

# 华南理工大学

## 《网络应用开发》课程实验报告

实验题目： 设计并实现一个电子商务网站的开发和在线部署

姓名： 黄鸿展 学号： 202230441138

班级： 计算机科学与技术 2 班

提交日期： 2024-12-6

代码托管地址： <https://github.com/Mmmouse404/web-work>

部署在线网站： [http://8.134.196.79:8080/javaweb mouse war exploded /](http://8.134.196.79:8080/javaweb%20mouse%20war%20exploded/)

测试账号口令： 用户账号:123456 密码:111 商家账号:1 密码:h1 管理密码:root

### 实验概述

#### 【实验题目】

设计并实现一个电子商务网站的开发和在线部署。

#### 【基本功能要求】

顾客：

用户的注册、登录、注销

展示产品列表

购买流程（浏览/查询->添加至购物车->付款->发送电子邮件确认收货）

可以查看订单状态和历史

销售：

商品目录的管理（包括最基本的增删改等操作）

订单管理、以及销售统计报表

客户管理、以及客户的 浏览/购买 日志 记录

#### 【实验环境和工具】

开发工具：IntelliJ IDEA 2024.1.6 数据库：MySQL 8.028

操作系统：Windows11, Alibaba Cloud Linux 3.2104 LTS 64 位

Web 服务器：阿里云 ECS 服务器(2 核 2GiB), tomcat 8.599

浏览器：Microsoft Edge 版本 131.0.2903.70(64 位)

编程框架：Java Servlet 后端、Jsp+ jQuery+ css 前端、tomcat 作为服务器

### 实验内容

#### 1. 系统设计：

## 1.1 数据库设计:

### 1.1.1 User 表

属性名称	属性类型	是否可为空	是否为主键
ID	int	否	是
NAME	varchar(100)	否	否
PASSWORD	varchar(100)	否	否
MONEY	Double	否	否
EMAIL	varchar(100)	是	否

### 1.1.2 Merchant 表

属性名称	属性类型	是否可为空	是否为主键
merchantname	varchar(100)	否	否
ID	int	否	是
PASSWORD	varchar(100)	否	否

### 1.1.3 Goodlists 表

属性名称	属性类型	是否可为空	是否为主键
id	int	否	是
goodname	varchar(100)	否	否
price	double	否	否
stock	int	否	否
kind	varchar(100)	否	否
merchantname	varchar(100)	否	否
description	varchar(100)	是	否

### 1.1.4 Cart 表

属性名称	属性类型	是否可为空	是否为主键
username	varchar(100)	否	否
goodname	varchar(100)	否	否
price	double	否	否
number	int	否	否
address	varchar(100)	否	否
merchantname	varchar(100)	否	否

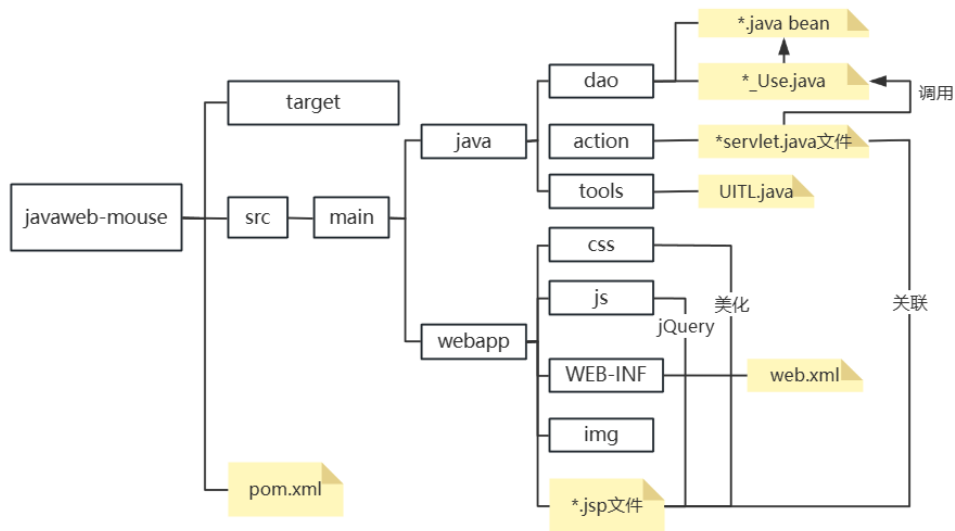
id	int	否	是
----	-----	---	---

1.1.5 Orders 表

属性名称	属性类型	是否可为空	是否为主键
ID	int	否	是
GOODNAME	varchar(100)	否	否
ADDRESS	varchar(100)	否	否
NUMBER	int	否	否
PRICE	double	否	否
MERCHANTNAME	varchar(100)	否	否
STATUS	varchar(100)	是	否

### 1.2 网站文件架构

网站的基本文件架构图如下：



其中：

1.2.1 target：文件存放源代码编译之后的结果

1.2.2 src/main：存放文件源代码

-java：存放后端 java 代码

-action：存放 servlet 文件，负责处理网页请求并转交给 service

-dao：设计需要用到的 java 类（数据结构），生成 SQL 语句，与数据库进行交互

-tools：设计 UTIL.java，设定账户与密码连接数据库

-webapp：存放前端代码

-img：存放背景图、logo 等图片

-js：ajax.js/json.js/jquery-3.4.1.min.js 支持处理 ajax、json、jQuery

-css：各个网页样式的 css 文件，美化网页

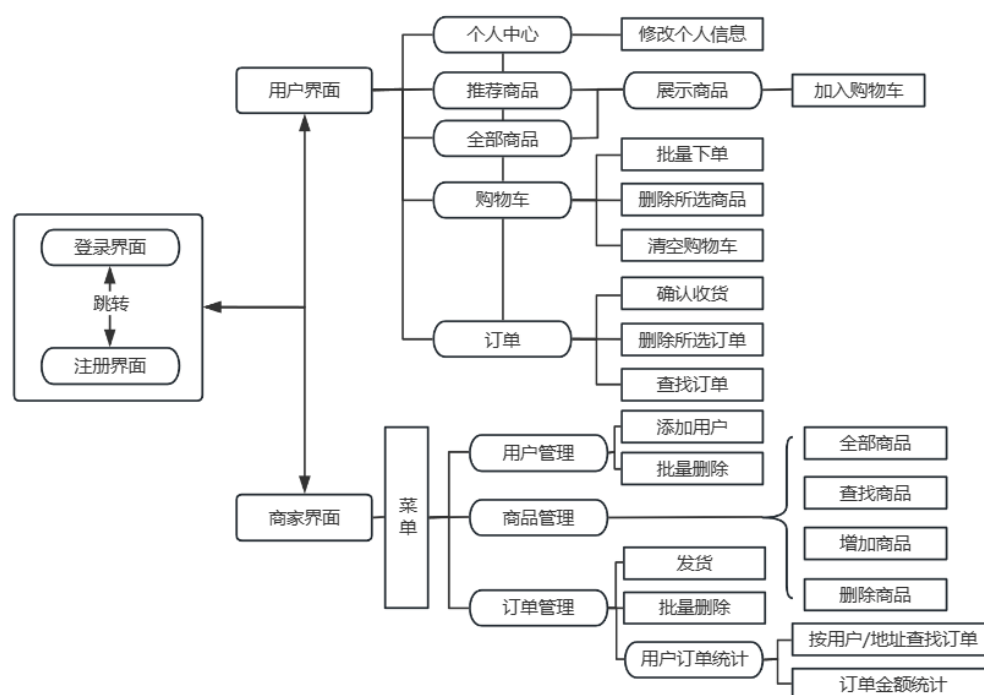
-WEB-INF：web.xml 设置 web 应用配置

—\*.jsp 文件：各个网页的设计

### 1.2.3 pom.xml：配置 Maven 项目所用依赖项

## 1.3 网站网页架构

网站网页架基本架构图如下：



## 1.4 基本功能实现

功能实现基本流程：（功能实现逻辑大体相似，调用函数）

- ①网页向 Servlet 发出请求
- ②Servlet 接收请求并转发给 Dao
- ③Dao 中通过\*\_Use 文件生成相应 SQL 语句与数据库进行交互并返回结果
- ④Servlet 得到结果，返回响应
- ⑤网页显示对应结果

### 1.4.1 注册、登录功能

用户与商家注册登录逻辑大体相似，按下用户/商家注册按钮弹出输入框，通过会话设置账号、名字、密码、邮箱（仅用户有）信息，并根据所选登录角色的不同跳转到用户界面或者商家界面。

### 1.4.2 用户功能

#### ① 个人中心

左侧展示个人信息，可以进行个人信息的输入（邮箱、密码），输入新信息后点击“修改”即可修改个人信息。通过增加金额的拖动条可以增加自己的金额，同时还可以直接点击对应的按钮跳转到对应的商品、购物车、订单页面。

#### ② 推荐商品

通过数据库的 goodlist 表直接随机选出 6 件商品横向排列到栏上展示图片和商品单价，名称。右上角通过增加金额的拖动条可以增加自己的金额，点入商品图片即可跳转到“展示商品”具体展示的页面。点击“刷新”按钮可以重新刷新商品。

### ③ 全部商品

通过数据库的 goodlist 表选择出所有商品，展示同推荐商品；有翻页功能，每页展示 10 件商品，点击商品图片即可展示商品的全部信息。通过增加金额的拖动条可以增加自己的金额。

### ④ 展示商品

通过“推荐商品”或“全部商品”点入时会通过 session 传入 Sid（商品编号），通过 Sid 查询对应商品的详细信息，以卡片信息展示到页面中。通过加减按钮可以更改购入物品数量，填写地址后点击“加入购物车”，通过操作数据库即可在 cart 中记录。

### ⑤ 购物车

购物车展示了 cart 中的所有商品购物车记录，含有批量删除（复选框）功能。选中复选框后即可在右侧看到“待结算的商品”，点击下单即可通过购物车的商品编号等信息，由数据库操作创建订单。还可以修改地址和数量栏改变订单的情况。在下单旁边还要清空购物车，点击即可删去该用户购物车内容。

### ⑥ 订单管理

展示了历史所有订单的情况（商品名，数量，总价，地址，用户名等），支持按地址和商品名进行查询。其中订单由“未发货”“已发货”“已收货”三个状态，如果订单状态为已发货即可按“收货”功能

## 1.4.3 商家功能

### ① 商品管理

商家登录后，即可看到商品管理的页面，展示了全部商品，可以按分类和名字进行搜索。可以点击“新增商品”，完成对商品信息的上传以后，通过数据库操作成功上传商品。点击修改商品即可让商品信息变为输入框，输入新信息点击修改即可完成修改。删除物品可以通过复选框完成批量删除。

### ② 订单管理

点击“菜单”-“商品管理”-“商品订单”即可看到该商家的商品订单所有情况。其中：“未发货”黄色，“已发货”为蓝色，“已收货”为绿色，“未发货”订单具有“发货”功能，点击发货即可修改订单状态，**同时利用 SMTP 协议对用户的邮箱提供邮件发货提醒。**可以通过复选框实现批量删除订单。

### ③ 用户订单统计

“菜单”-“用户管理”-“用户订单日志”即可看到以用户进行查询的订单情况，统计了每个用户订单共计总消费额。而点击用户的表格即可看到每个订单的具体情况，包括是否发货，也可以在此页面实现发货。可以根据用户名或地址进行指定用户订单查询。

### ④ 用户管理

“菜单”-“用户管理”-“用户信息”，输入“root”密码即可登录，查看用户的所有信息，包括密码（因此需要管理员密码 root，和普通商家权限分离），可以新增用户，也可进行用户的批量删除。

## 2. 代码实现

### 2.1 基本数据结构

基本用到的 java 类都存在 Dao 中，数据结构与数据库的对应，varchar 改为用 String，包含了 Get 和 Set 方法，以 User 为例：

```
public class User { 10个用法
    int id; //账号 2个用法
    String name; //用户名 2个用法
    String password; //密码 2个用法
    double money; 2个用法
    String email; //邮箱 2个用法
    public int getId() { return id; }

    public void setId(int id) { this.id = id; }
```

而 User\_Use 则包含了 User 类的使用方法，主要是以数据库的增、删、查、改操作为主。

```
public class User_Use {
    private static int number1; //用于账号密码的匹配 3个用法
    private static int number2; 3个用法

    public static void in(String loginname,String loginid,String loginpass,double money,String email) throws SQLException {
        Connection con= UTIL.getCon(); //建立 连接
        String sql= "insert into users (NAME, ID, PASSWORD, MONEY, EMAIL)" +
            "values(?, ?, ?, ?, ?)";
        //设置变量，用?占位符 等下会传入值过来
        PreparedStatement gg=con.prepareStatement(sql);
        // 要在执行前赋值 第一个? 序号是1
        gg.setString( parameterIndex: 1,loginname);
        gg.setString( parameterIndex: 2,loginid);
        gg.setString( parameterIndex: 3,loginpass);
        gg.setDouble( parameterIndex: 4,money);
        gg.setString( parameterIndex: 5,email);
        gg.executeUpdate(); //执行,返回结果是操作了几条数据
    }

    public static boolean selectName(String name) throws SQLException { //查询name 1个用法
```

涉及的数据结构与数据库基本类似，并以\*\_Use. java 作为数据库操作方法类，下面给出各个数据结构：

```
public class Cart { 10个用法
    int id; 3个用法
    String goodname; //商品名字 2个用法
    String username; //用户名字 2个用法
    String address; //地址 2个用法
    int number; //数量 2个用法
    String price; //价格总和 2个用法
    String merchantname; 2个用法
```

```
public class Goodlist {
    String id; //商品编号 2个用法
    String goodname; //商品名字 2个用法
    String image; //商品图片 2个用法
    String kind; //商品类别 2个用法
    String price; //商品价格 2个用法
    String stock; //库存 2个用法
    String merchantname; 2个用法
    String description; 2个用法
```

```
public class Merchant { 0个用法
    private int id; //商家ID 2个用法
    private String merchantname; //商家名称 2个用法
    private String password; //密码 2个用法
```

```
public class Order { 34个用法
    int id; //订单编号，自增长 3个用法
    String goodname; //商品名字 2个用法
    String username; //用户名字 2个用法
    String address; //地址 2个用法
    int number; //数量 2个用法
    String price; //价格总和 2个用法
    String merchantname; 2个用法
    String status; 2个用法
```

## 2.2 基本代码流程

以商品加入购物车为例子讲解代码实现的基本流程：

```

<tr>
  <td>
    <div class="goods_num_wrap">
      购买数量: <span id="i1" style="text-align: center;"></span>
      <input type="text" value="1" class="goods_num" id="number">
      <span id="i2" style="text-align: center;">+</span>
    </div>
  </td>
</tr>
<tr>
  <td>
    <div class="address-info">
      地址: <input type="text" placeholder="请填写地址, 例: 6#505" id="dz">
    </div>
  </td>
</tr>
<tr>
  <td>
    <div class="button-container">
      <input type="button" value="加入购物车" class="add-cart-button" onclick="addToCart()" /> <!-- 添加到购物车 -->
      <input type="button" value="返回" class="add-cart-button" onclick="history.back()" /> <!-- 返回按钮 -->
    </div>
  </td>
</tr>
</table>

```

首先通过 jsp 和 css 创建一个页面的输入框，通过 Goodlist 的 Get 方法得到对应商品的信息（图片，名字，价格等），通过 input 标签输入框引导用户输入地址，通过加号减号进行数量的删减。而下方的<script>脚本标签则保证了加减号更新时物品的购入数量改变，同时通过查询物品的库存限制加减号最多到库存上限，并计算购入的总金额，传入#good\_total 标签，更新总价。

```

$(function () {
  function calc() {
    var price = parseFloat($("#goods_price").text()); // 获取单价
    var quantity = parseInt($("#number").val()); // 获取数量
    var stock = parseInt($("#stock").text()); // 获取库存
    var total = (price * quantity).toFixed(2); // 计算总价
    $("#good_total").text(total); // 显示总价

    // 限制数量不能超过库存
    if (quantity > stock) {
      quantity = stock; // 如果超过库存，将数量设置为库存
      $("#number").val(quantity); // 更新输入框
      alert("已达到库存上限，最多只能购买 " + stock + " 件。");
    }

    // 更新加号按钮的状态
    updateAddButtonState(quantity, stock);
  }

  // 更新加号按钮的状态
  function updateAddButtonState(quantity, stock) {
    if (quantity >= stock) {
      $("#i2").addClass("unavailable"); // 添加变暗样式
    } else {
      $("#i2").removeClass("unavailable"); // 移除样式
    }
  }
});

```

```

$("#i2").click(function () {
  var num = parseInt($("#number").val());
  var stock = parseInt($("#stock").text()); // 获取库存
  num++; // 自增
  // 限制数量不能超过库存
  if (num > stock) {
    num = stock; // 如果超过库存，将数量设置为库存
    alert("已达到库存上限，最多只能购买 " + stock + " 件。");
  }
  $("#number").val(num);
  calc(); // 重新计算总价
});

// 手动输入数量计算
$("#number").on('input', function () {
  calc(); // 每次输入后重新计算总价
});

```

获取了购物的信息以后，通过 `addToCart()` 函数将所有的购物信息合并在一个 `param` 中，并通过 `ajax` 方法向 `addToCartServlet` 传入 `Post` 请求，传入 `param` 数据，并等待响应。

```
function addToCart() {
    var goodname = $("#tt").text(); // 商品名称
    var username = "<%=session.getAttribute('username')%>"; // 用户名称
    var address = $("#dz").val(); // 用户填写的地址
    var number = $("#number").val(); // 商品数量
    var price = $("#good_total").text(); // 商品总价
    var merchantName = $("#merchantname").text(); // 商家名称
    var goodId = $("#good_id").text(); // 商品ID
    var description = $("#description").text(); // 商品描述
    // 验证输入信息是否合法
    if (!goodname || !username || !address || !number || !price || !merchantName) {
        alert("请确保所有信息都填写正确！");
        return;
    }
    var param = "goodname=" + encodeURIComponent(goodname)
        + "&username=" + encodeURIComponent(username)
        + "&address=" + encodeURIComponent(address)
        + "&number=" + encodeURIComponent(number)
        + "&price=" + encodeURIComponent(price)
        + "&merchantname=" + encodeURIComponent(merchantName) // 注意大小写
        + "&goodId=" + encodeURIComponent(goodId)
        + "&description=" + encodeURIComponent(description);
    // 发送请求到 addToCart Servlet
    $.ajax({
        url: "addToCart", // 后端处理的 URL
        type: "POST",
        data: param,
        success: function(response) {
            console.log("服务器响应:", response); // 添加日志
            if (response.trim() === "success") { // 使用 trim() 去除首尾空格
                alert("成功加入购物车！");
            } else {
                alert(response); // 显示错误内容
            }
        }
    });
}
```

在 `Post` 请求后，`Servlet` 响应请求并执行 `doPost()` 方法，从 `request` 里通过 `getParameter` 方法得到各个属性，并调用 `Cart` 中的 `Set` 方法设定一个 `Cart` 对象，最后通过 `Cart_Use` 中的 `AddToCart()` 方法执行数据库操作，插入一个 `Cart` 对象，最终通过 `write` 和 `setStatus()` 方法设置响应状态并写回内容，`addToCart()` 函数接收响应并根据响应是否成功给出信息。



```

@WebServlet("/addToCart")
public class addToCartServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");response.setContentType("text/html;charset=utf-8");
        String goodName = request.getParameter( name: "goodname");
        String username = request.getParameter( name: "username");
        String address = request.getParameter( name: "address");
        String numberStr = request.getParameter( name: "number");
        String price = request.getParameter( name: "price");
        String merchantname = request.getParameter( name: "merchantname");
        int number;
        try {
            number = Integer.parseInt(numberStr);
            if (number <= 0) {
                response.setStatus(HttpServletResponse.SC_BAD_REQUEST);
                response.getWriter().write( "数量必须大于0");
                return;}
        } catch (NumberFormatException e) {
            response.setStatus(HttpServletResponse.SC_BAD_REQUEST);
            response.getWriter().write( "数量格式不正确");
            return;}
        // 创建一个 Cart 对象
        Cart cartItem = new Cart();
        cartItem.setGoodName(goodName);
        cartItem.setUserName(username);
        cartItem.setAddress(address);
        cartItem.setNumber(number);
        cartItem.setPrice(price);
        cartItem.setmerchantname(merchantname); // Set merchantname
        try {
            Cart_Use.addToCart(cartItem); // 将商品添加到购物车
            response.setStatus(HttpServletResponse.SC_OK); // 设置响应状态为成功
            response.getWriter().write( "成功加入购物车！"); // 发送成功消息
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
// 增加商品到购物车
public static void addToCart(Cart cartItem) throws SQLException { 1个用法
    Connection con = UTIL.getCon();
    String sql = "INSERT INTO cart (goodname, username, address, number, price, merchantname) VALUES (?";
    PreparedStatement ps = con.prepareStatement(sql);
    ps.setString( parameterIndex: 1, cartItem.getGoodName());
    ps.setString( parameterIndex: 2, cartItem.getUserName());
    ps.setString( parameterIndex: 3, cartItem.getAddress());
    ps.setInt( parameterIndex: 4, cartItem.getNumber());
    ps.setString( parameterIndex: 5, cartItem.getPrice());
    ps.setString( parameterIndex: 6, cartItem.getmerchantname());
    ps.executeUpdate();
}

```

由于具体代码太多，且操作流程与上述相似，这里将给出其他关键代码的说明：

## 2.3 关键功能代码说明

### 2.3.1 Dao

User：用户类，User\_Use:用户操作类，可以根据用户 id 或名字在数据库中更新信息或删除，可以在数据库中添加新用户

Merchant：商家类，Merchant\_Use：商家操作类，可以根据商家的 id 找到对应商家的信息，可以添加/删除商家信息

Goodlist：商品类，包含商品姓名、单价、图片路径等信息，

Goodlist\_Use:商品操作，可以进行商品的增删改查，主要以传入 Id 或 Goodname 作为查询参数，进行数据库操作。getRandomGoods() 还可以随机查询对应数量的商品。

Cart：购物车类，主要存放加入购物车的信息，Cart\_Use：负责对应用户的购物车信息，可以根据 Goodlist 的 Id 进行增删改查，还可以向 Order 传递对应的用户名字、商品地址参数。

Order：订单类，主要收集用户信息、物品信息、商家信息（只收集主键及地址），其他部分通过调用其他 Use 类进行展示。可以根据 User 中的 Email 通过 javax.mail 中的方法发送邮件。

### 2.3.2 Action

addGoodServlet: 商家增加商品  
addOrderServlet: 用户增加订单（购物车结算）  
addToCartServlet: 指定商品加入到购物车  
cartActionServlet: 购物车操作，主要负责删除与清空购物车  
goodBatchDeleteServlet: 商家批量删除商品  
increaseMoneyServlet: 用户增加金额  
loginServlet: 用户登录  
logoutServlet: 用户与商家注销  
merchantLoginServlet: 商家登录  
merchantRegisterServlet: 商家注册  
orderBatchDeleteServlet: 批量删除订单（商家用户均可）  
registerServlet: 用户注册  
setRootSessionServlet: 商家访问用户信息，开启管理员权限  
shipOrderServlet: 修改订单状态（发货-收货）  
updateGoodServlet: 商家修改商品信息  
updateUserInfoServlet: 修改用户信息  
userBatchDeleteServlet: 用户批量删除

### 2.3.3 Tools

UTIL.java: 创建了一个 Connection 对象，加载好数据库驱动后连接云服务器的 mysql 数据库，通过给定的用户名登录并保持连接

### 2.3.4 Jsp

index.jsp: 索引界面，同登陆界面  
register.jsp: 注册界面，包含用户与商家注册  
registersucc.jsp: 注册成功，等待一秒跳转到登陆界面  
login.jsp: 登陆界面，包含用户登录与商家登录  
loginFail.jsp: 登陆失败，显示错误原因，等待一秒回到登陆界面  
loginSuccess.jsp: 登陆成功，等待一秒直接显示商品信息  
userProfile.jsp: 个人中心界面，修改个人信息，可以跳转到其他界面  
mainFrame.jsp: 推荐商品界面，随机推荐 6 个商品  
allShop.jsp: 所有商品界面，可以翻页，可以按分类或名字搜索（模糊）  
mouseShop.jsp: 具体商品界面，显示商品信息，可以加入购物车或返回  
cart.jsp: 购物车界面，可以勾选商品修改数量地址下单、删除商品  
myOrders.jsp: 用户个人订单界面，修改订单状态、删除订单  
manageGood.jsp: 商品展示界面，可以跳转到商品的增删改查界面  
insertGood.jsp: 商品添加界面，填入信息后可以上传商品  
editGood.jsp: 修改商品界面，弹出所有商品，输信息即可修改，可以查询  
manageOrder.jsp: 订单管理界面，可以修改订单状态或查询、删除  
manageUserOrder.jsp: 用户订单统计页面，展示各个用户的订单详情，总金额，点击用户可以查看该用户的所有订单  
manageUser.jsp: 用户管理界面（仅管理员），登陆后展示用户信息，可以增加或删除用户信息

header.jsp: 顶部栏, 便于跳转各个界面

sidebar.jsp: 侧边栏, 设计菜单, 可以跳转到各个界面

## 2.4 代码部署

① 在 github 上注册一个账号

Public profile

Name

Nezumy

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at a

Public email

Select a verified email to display

You have set your email address to private. To toggle email privacy, go to [email settings](#) and uncheck "email address private."

Bio

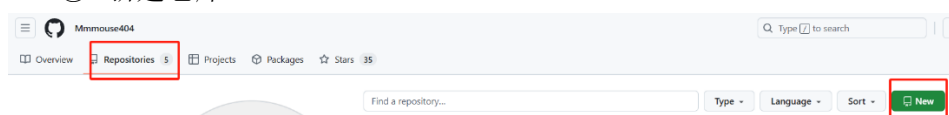
South China University Of Technology in Computer Science

You can @mention other users and organizations to link to them.

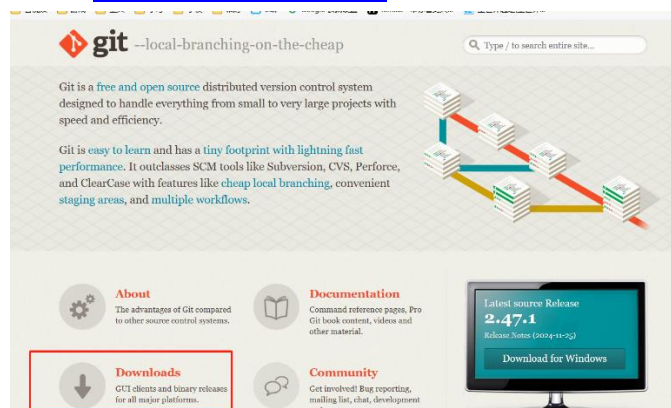
Pronouns

Don't specify

② 新建仓库



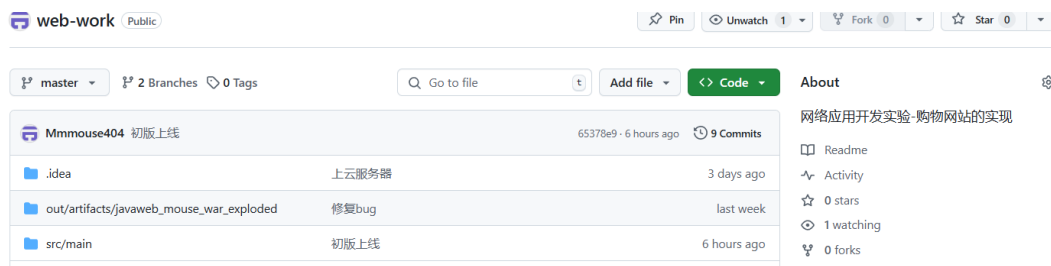
③ 安装 git <https://git-scm.com/>



④ 利用 git 初始化, 上传代码

```
先找到你放置项目的目录, 打开 git
cd ["你放置项目的目录"]
git init
git add .
git add README.md
git commit -m "注释语句"
git remote add origin https://github.com/zlxzlxzx/Test.git
"https://github.com/zlxzlxzx/Test.git"
为你对应的 github 仓库地址, 可以在 Code-Local-Clone 中找到
git push -u origin master
如果冲突, 可以将 -u 改为 -f
```

⑤ 可以在 github 中看到你上传的项目文件:



我的 github 项目链接: <https://github.com/Mmmouse404/web-work>

### 3. 功能测试:

#### 3.1 用户

##### 3.1.1 登录与注册



首先进入登陆界面, 显示用户登录与商家登录界面。由于未注册, 我们先点击注册按钮, 跳转到注册页面:



我们注册账号为 123123, 用户名为 123, 密码为 123, 邮箱为 [cshhz@mail.scut.edu.cn](mailto:cshhz@mail.scut.edu.cn) 的用户, 跳转到如下界面:

**用户:123 注册成功!**

正在前往登录页面.....

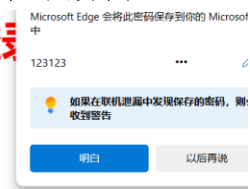




如果输入密码错误，则会显示错误原因回到登陆界面：

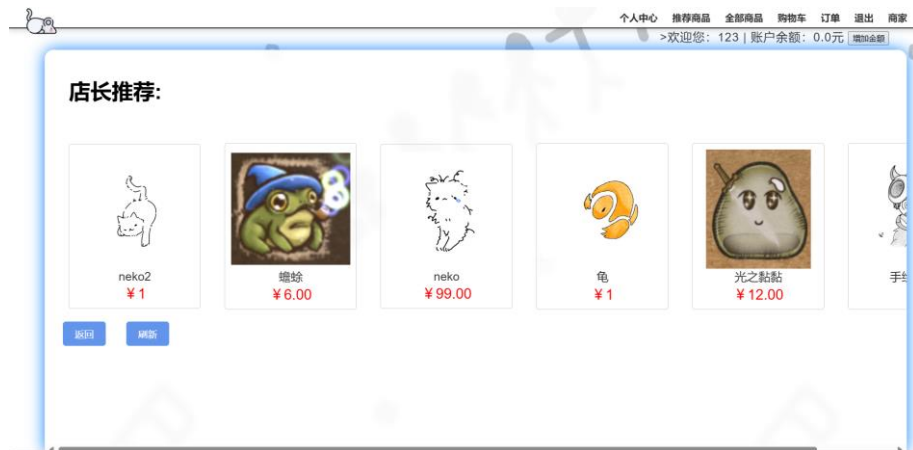
**用户:123 密码输入错误 登录失败**

正在返回登录页面.....



### 浏览与搜索

用户登录，如果成功则进入用户界面：



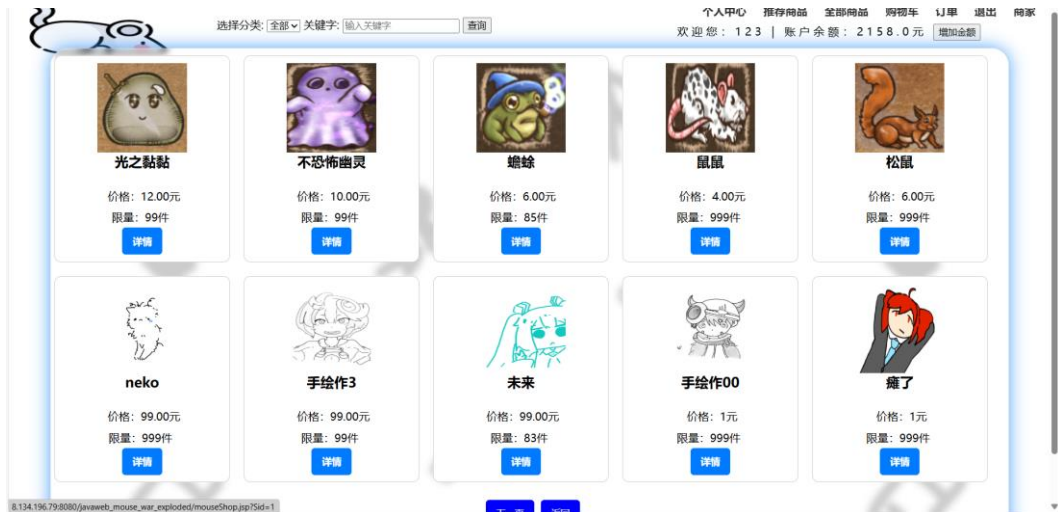
点击右上角的增加金额，拖动下方圆标，提交即可增加对应金额：

>欢迎您: 123 | 账户余额: 2158.0元 [增加金额](#)

增加金额

[提交](#)

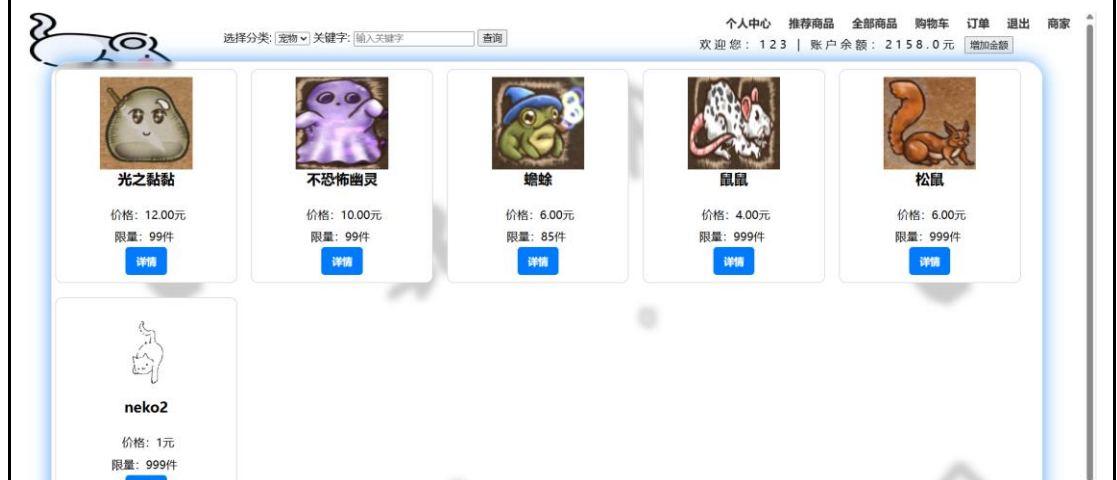
点击上方”全部商品”，即可看到全部商品，可以翻页：



点入具体的商品图片，跳转到具体商品展示页面



在上方的搜索栏，可以输入名字/选择分类进行搜索（模糊搜索，返回所有）



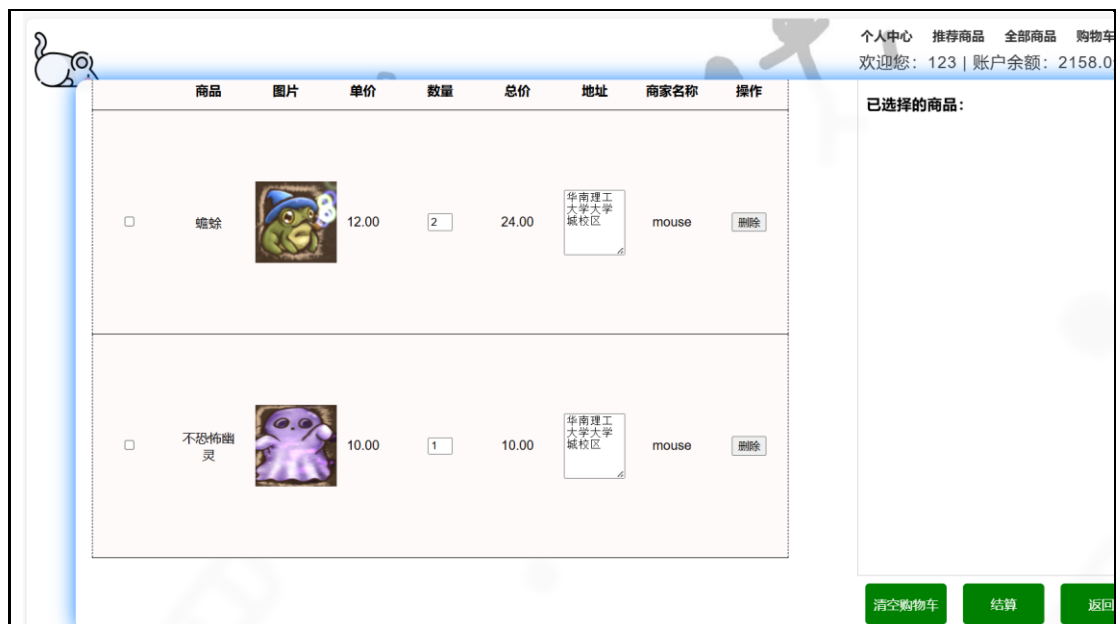
### 3.1.2 选购与结算

可以按加减调整数量（不超过库存），会得到对应总价。填写地址后即可加入购物车



点击“购物车”按钮，即可跳转到购物车界面。





### 3.1.3 下单与收货

购物车中展示了选择的商品，可以修改数量地址，勾选后将显示下单商品：



按下结算按钮，即可创建订单，按下删除/清空购物车按钮，对应商品将在购物车中删除：



点击“订单”按钮，即可看到自己下的单：





如果订单状态有更新，显示为“已发货”，则会显示收货按钮，按下后将更新订单状态会从“已发货”改为“已收货”。

## 我的订单

选择	商品名称	地址	数量	价格	商家名称	订单状态	
<input type="checkbox"/>	不恐怖幽灵	华南理工大学大学城校区	1	10.00	mouse	已发货	收货

删除所选订单

返回

### 3.1.4 个人信息修改

点击个人中心，即可进入个人中心界面，输入新邮箱/新地址即可进行修改。

个人中心

用户信息

用户名:

123

邮箱:

cshhz@mail.scut.edu.cn

新密码:

...

更新信息

账户余额:

2148.0

元

增加金额 | 信息更新成功!

推荐商品

全部商品

购物车

订单



## 3.2 商家

### 3.2.1 登陆与注册

注册流程同用户。我们注册账号为 1234，用户名为好耗子，密码为 1234 的商家，注册成功后登录，跳转到如下界面：

商家: 好耗子 登录成功

正在进入.....



Microsoft Edge 会将此密码保存到你的 Microsoft 帐户中

1234

编辑

如果在联机设备中发现保存的密码, 则会收到警告

明白

以后再说

首页

管理员 用户登录 退出

菜单

首页 商品管理

选择分类: 全部 关键字: 关键词 查询

新增商品 修改商品 批量删除

<input type="checkbox"/> 全选	编号	商品类别	商品名称	商品价格	商品库存	商家名称	商品描述	商品图片
总计0 条								

3.2.2 商品管理（增删改查）

点击菜单-商品管理-新增商品（或直接新增商品）即可跳转到增加商品页面，输入信息后即可添加商品。


小卖部上架商品

\* 商品编号:

自动生成

\* 选择图片:

选择文件 | b...



\* 商品类别:

画画

\* 商品名称:

祈福

\* 商品价格:

18.00

\* 商品库存:

999

\* 商家名称:

mouse

\* 商品描述:

放空大脑, 祝愿手


上架

返回

首页 > 商品管理

选择分类: 全部 关键字: 查询

新增商品 修改商品 批量删除

<input type="checkbox"/> 全选	编号	商品类别	商品名称	商品价格	商品库存	商家名称	商品描述	商品图片
<input type="checkbox"/>	1	宠物	光之黏黏	12.00元	99	mouse	111	
<input type="checkbox"/>	2	宠物	不恐怖幽灵	10.00元	98	mouse	222	
<input type="checkbox"/>	3	宠物	蟾蜍	6.00元	85	mouse	null	
<input type="checkbox"/>	23	画画	祈福	18.00元	999	mouse	放空大脑，祝愿我们抵达群星	

商品管理界面可以根据分类和名字进行模糊搜索：



选择分类: 全部 关键字: 光 查询

新增商品 修改商品 批量删除

<input type="checkbox"/> 全选	编号	商品类别	商品名称	商品价格	商品库存	商家名称	商品描述	商品图片
<input type="checkbox"/>	1	宠物	光之黏黏	12.00元	99	mouse	111	

选择分类: 画画 关键字: 查询

新增商品 修改商品 批量删除

<input type="checkbox"/> 全选	编号	商品类别	商品名称	商品价格	商品库存	商家名称	商品描述	商品图片
<input type="checkbox"/>	6	画画	neko	99.00元	999	mouse	null	
<input type="checkbox"/>	19	画画	手绘画00	1元	999	mouse	手绘画11	

点击“修改商品”，将弹出所有商品信息，输入新信息，按下修改，即可修改。

小卖部修改商品 [返回](#)

选择分类: 全部 关键字:

	商品名称	商品类别	商品价格	商品库存	商家名称	商品描述	选择图片	操作
1	<input type="text" value="光之黏黏"/>	<input type="text" value="宠物"/>	<input type="text" value="12.00"/>	<input type="text" value="99"/>	<input type="text" value="mouse"/>	<input type="text" value="111"/>	<div>选择文件 未...</div> 	<a href="#">修改</a>
2	<input type="text" value="不恐怖幽灵"/>	<input type="text" value="宠物"/>	<input type="text" value="10.00"/>	<input type="text" value="98"/>	<input type="text" value="mouse"/>	<input type="text" value="222"/>	<div>选择文件 未...</div> 	<a href="#">修改</a>

选择图片 未...

勾选商品框，点击“批量删除”，即可删除对应商品。

新增商品 修改商品 批量删除

<input type="checkbox"/> 全选	编号	商品类别	商品名称	商品价格	商品库存	商家名称	商品描述	商品图片
<input type="checkbox"/>	1	宠物	光之黏黏	12.00元	99	mouse	111	
<input checked="" type="checkbox"/>	2	宠物	不恐怖幽灵	10.00元	98	mouse	222	

目录 商品管理

选择分类: 全部 关键字:  [查询](#)

新增商品 修改商品 批量删除

<input type="checkbox"/> 全选	编号	商品类别	商品名称	商品价格	商品库存	商家名称	商品描述	商品图片
<input type="checkbox"/>	1	宠物	光之黏黏	12.00元	99	mouse	111	
<input type="checkbox"/>	3	宠物	蟾蜍	6.00元	85	mouse	null	
<input type="checkbox"/>	4	宠物	鼠鼠	4.00元	999	mouse	null	

### 3.2.3 订单管理

点击菜单-商品管理-商品订单，即可看到所有订单信息。

菜单

用户管理

商品管理

全部商品

添加商品

修改商品

删除商品

商品订单

订单管理

新增订单 批量删除

<input type="checkbox"/> 全选	订单编号	订单商品	订单人	地址	数量	总价	状态	操作
<input type="checkbox"/>	3	鼠标	hhz2	华南理工大学大学城校区	2	8.00元	已发货	
<input type="checkbox"/>	9	鼠标	hhz1	South China University of Technology	2	8.00元	已收货	
<input type="checkbox"/>	11	键盘	hhz1	华南理工大学大学城校区	3	18.00元	已收货	
<input type="checkbox"/>	12	键盘	hhz1	华南理工大学大学城校区	4	18.00元	已收货	
<input type="checkbox"/>	13	鼠标	hhz1	South China University of Technology	2	8.00元	已发货	
<input type="checkbox"/>	14	键盘	hhz1	华南理工大学大学城校区	2	12.00元	null	
<input type="checkbox"/>	16	光之黏黏	hhz1	华南理工大学大学城校区	3	36.00元	null	
<input type="checkbox"/>	17	键盘	hhz1	华南理工大学大学城校区	1	6.00元	null	
<input type="checkbox"/>	24	鼠标	322	23123	1	4.00元	null	

为实现用户权限分离，将新增订单功能剔除，只留下批量删除。如果订单状态为“未发货”，将会弹出“发货”按钮，修改状态通知根据对应用户的邮箱发出商品发货邮件：

<input type="checkbox"/>	34	不恐怖幽灵	123	华南理工大学大学城校区	1	10.00元	未发货	发货
<input type="checkbox"/>	34	不恐怖幽灵	123	华南理工大学大学城校区	1	10.00元	已发货	

收件箱 您的订单已... x 您的订单已... x

回复 回复全部 转发 移动到 标记为 更多 删除

您的订单已发货 18126422610@163.com 发送给 黄鸣震 2024-12-07 00:07:51

尊敬的用户，您的订单（订单编号：34）已成功发货，感谢您的购买！<br>商家名称：mouse

3.2.4 用户订单统计

点击用户管理-用户订单日志，即可看到不同用户订单统计情况：

用户订单管理 (点击用户可查看订单详情)

搜索用户名

用户名称	订单数量	总消费额
hhz1	17	832.0元
322	4	1489.0元
耗子	0	0.0元
111	0	0.0元
123	1	10.0元

点击对应用户单元格，即可弹出详细用户订单，也可以发货：

订单编号	订单商品	地址	数量	状态	总价	操作
9	鼠鼠	South China University of Technology	2	已收货	8.00元	
11	蟾蜍	华南理工大学大学城校区	3	已收货	18.00元	
12	蟾蜍	华南理工大学大学城校区	4	已收货	18.00元	
13	鼠鼠	South China University of Technology	2	已发货	8.00元	
14	蟾蜍	华南理工大学大学城校区	2	未发货	12.00元	发货
15	未来	华南理工大学大学城校区	2	未发货	198.00元	发货
16	光之黏黏	华南理工大学大学城校区	3	未发货	36.00元	发货
17	蟾蜍	华南理工大学大学城校区	1	未发货	6.00元	发货
18	未来	华南理工大学大学城校区	2	未发货	198.00元	发货
26	蟾蜍	华南理工大学大学城校区	1	未发货	6.00元	发货
27	光之黏黏	华南理工大学大学城校区	3	未发货	36.00元	发货

也可以输入用户名进行模糊查询：

用户订单管理 (点击用户可查看订单详情)

3

用户名称	订单数量	总消费额
322	4	1489.0元
123	1	10.0元

订单编号	订单商品	地址	数量	状态	总价	操作
21	未来	2535	5	未发货	495.00元	发货
22	未来	2535	5	未发货	495.00元	发货
23	未来	2535	1	未发货	495.00元	发货
24	鼠鼠	23123	1	未发货	4.00元	发货

订单编号	订单商品	地址	数量	状态	总价	操作
34	不恐怖幽灵	华南理工大学大学城校区	1	已发货	10.00元	

3.2.5 用户管理（管理员特权）

点击用户管理-用户信息，会弹出登录界面：

请输入管理员密码(如登录成功，关闭即可)

提交

关闭

输入登录密码（root），即可进入界面，得到用户的信息：  
否则将会跳转到管理商品界面。

首页 > 用户信息

批量删除 新增用户

<input type="checkbox"/> 全选	用户名	账号	密码
<input type="checkbox"/>	hhz1	123456	111
<input type="checkbox"/>	322	233332	233
<input type="checkbox"/>	耗子	123789	h1
<input type="checkbox"/>	111	111111	11
<input type="checkbox"/>	123	123123	123

总计5条 1/1 页

可以新增用户，也可以批量删除用户。（无法指定邮箱）

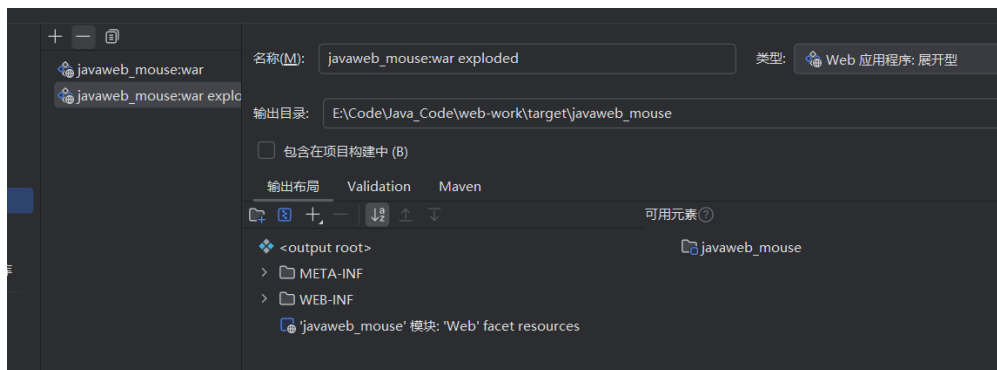
### 新增用户

用户名	账号
密码	
提交	关闭

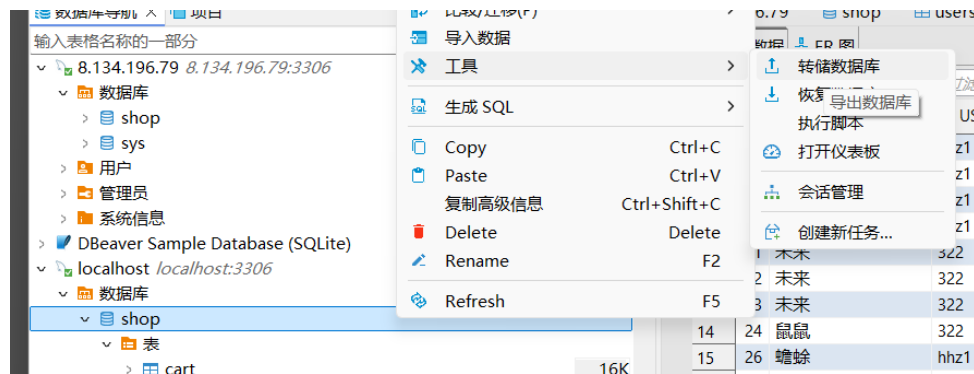
## 4. 应用部署

### 4.1 文件准备

在 idea 中准备好项目工件（远程需要 war exploded）



可以在本地准备好数据库直接导出 sql 文件，以便云端恢复。



### 4.2 服务器准备

#### 4.2.1 服务器购买搭建

登录阿里云，进行学生认证：

<https://myaccount.console.aliyun.com/basic-info?open=student>

完成学生认证后可以在此处领取代金券购买服务器：

<https://university.aliyun.com/>



地区选择华南，操作系统我选用的是 Alibaba Cloud Linux 3.2104 LTS 64 位。

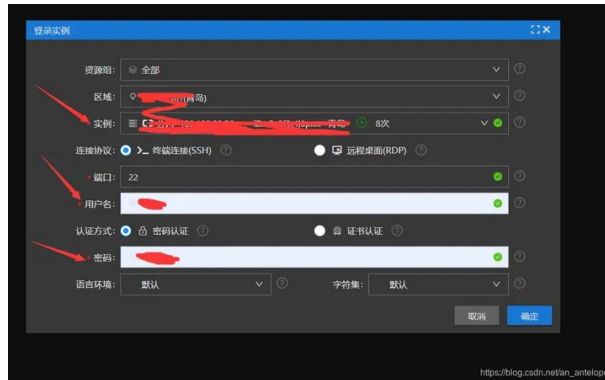
购买以后还需购买弹性公网 IP



登陆以后需要通过远程连接，初次登录可能需要设置系统密码。







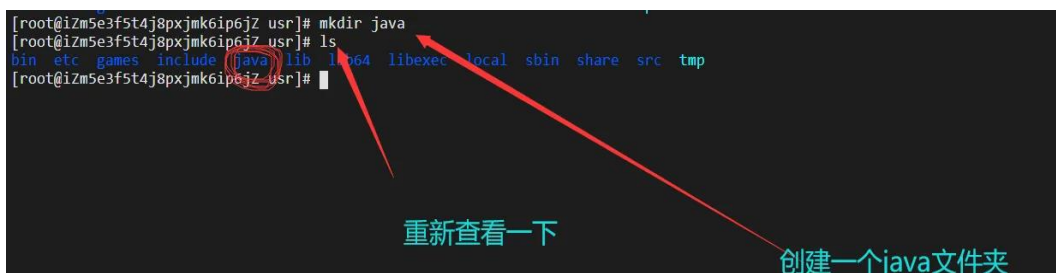
#### 4.2.2 所需工具配置

采用工具 FileZilla, 通过远程端口 21 登录连接到操作系统, 以便文件传输:



##### ① 安装 tomcat:

- 1: 登录阿里云, 用 File Zilla 连接阿里云
- 2: 创建 jdk 存放的位置文件夹



cd /usr mkdir java

- 3: 将 jdk8-linux-64.tar.gz 包上传到 java 文件夹中



#### 4: 解压 jdk8-linux-64.tar.gz

```
tar -zxvf jdk8-linux-64.tar.gz
```

```
[root@izM5e3f5t4j8pxjmk6ip6jZ java]# tar -zxvf jdk8-linux-64.tar.gz
```

解压这个tar文件

#### 5: 解压完成后将解压后的文件改名字方便使用

mv 解压出来的文件名字 要改成什么

```
mv jdk1.8.0_212 jdk1.8
```

```
[root@izM5e3f5t4j8pxjmk6ip6jZ java]# mv jdk1.8.0_212 jdk8
```

为了方便之后操作，给他改一个名字

#### 6: 配置 jdk 的环境变量

```
vim /etc/profile
```

```

> 2. root@izm5e3f5t4j8pxjmk6ip6jz:/usr/java X
pathmunge /usr/local/sbin
else
pathmunge /usr/local/sbin after
pathmunge /usr/sbin after
fi

HOSTNAME="/usr/bin/hostname >/dev/null"
HISTSIZE=1000
if [ "$HISTCONTROL" = "ignorespace" ] ; then
export HISTCONTROL=ignoreboth
else
export HISTCONTROL=ignoredups
fi

export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE HISTCONTROL

# By default, we want umask to get set. This sets it for login shell
# Current threshold for system reserved uid/gids is 200
# You could check uidgid reservation validity in
# /usr/share/doc/setup-*/uidgid file
if [ $UID -gt 199 ] && [ "/usr/bin/id -gn" = "/usr/bin/id -un" ]; then
umask 002
else
umask 022
fi

for i in /etc/profile.d/*.sh /etc/profile.d/sh.local ; do
if [ -r "$i" ] ; then
if [ "${-#*i}" != "$-" ] ; then
. "$i"
else
. "$i" >/dev/null
fi
fi
done

unset i
unset -f pathmunge

```

打开的文件是这个样子的

到最后输入i

进入后 敲 i 进行编辑 加入下列四条语句

```

export JAVA_HOME=/usr/java/jdk8
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH

```

(注: export JAVA\_HOME=jdk 的安装路径)

保存并退出 ESC 退出编辑; wq 保存并退出文档

```

umask 002
else
umask 022
fi

for i in /etc/profile.d/*.sh /etc/profile.d/sh.local ; do
if [ -r "$i" ] ; then
if [ "${-#*i}" != "$-" ] ; then
. "$i"
else
. "$i" >/dev/null
fi
fi
done

unset i
unset -f pathmunge
export JAVA_HOME=/usr/java/jdk8
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
:wq

```

按esc退出编辑模式, 再按冒号输入wq, 代表退出文档并保存

运行改过的文件即可

source /etc/profile

```
7-2 root@izm5e3f5t4j8pxjmk6ip6jz:/usr/java
[root@izm5e3f5t4j8pxjmk6ip6jz java]# source /etc/profile
[root@izm5e3f5t4j8pxjmk6ip6jz java]# java -version
java version "1.8.0_212"
Java(TM) SE Runtime Environment (build 1.8.0_212-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.212-b10, mixed mode)
[root@izm5e3f5t4j8pxjmk6ip6jz java]#
```

就可以查看版本，看看安装是否成功

执行一下我们刚刚改过的文件

## ② 安装 TomCat

1: 连接服务器，连接 File Zilla (将压缩包传到 linux 服务器上)

2: 创建 tomcat 存放的文件夹

```
cd /usr/local mkdir tomcat
```

```
[root@izm5e3f5t4j8pxjmk6ip6jz ~]# cd /usr/local
[root@izm5e3f5t4j8pxjmk6ip6jz local]# ls
aegis bin etc games include lib lib64 libexec sbin share src
[root@izm5e3f5t4j8pxjmk6ip6jz local]# mkdir tomcat
[root@izm5e3f5t4j8pxjmk6ip6jz local]# ls
aegis bin etc games include lib lib64 libexec sbin share src tomcat
[root@izm5e3f5t4j8pxjmk6ip6jz local]#
```

创建文件夹tomcat

3: 用 File Zilla 将 tomcat 上传到服务器中

4: 上传完成后解压《apache-tomcat-8.5.20.tar》并改名方便使用

用

/\*解压\*/ `tar -zxvf apache-tomcat-8.5.20.tar`

/\*改名\*/ `mv apache-tomcat-8.5.20 tomcat8.5`

```
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]# ls
apache-tomcat-8.5.20.tar.gz
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]# tar -zxvf apache-tomcat-8.5.20.tar.gz
```

解压

```
apache-tomcat-8.5.20/bin/version.sh
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]# ls
apache-tomcat-8.5.20 apache-tomcat-8.5.20.tar.gz
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]#
```

再次查看会多一个解压过后的文件夹

```
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]# mv apache-tomcat-8.5.20 tomcat8.5
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]# ls
apache-tomcat-8.5.20.tar.gz tomcat8.5
[root@izm5e3f5t4j8pxjmk6ip6jz tomcat]#
```

进行改名方便以后使用

4: 配置 tomcat 并运行 tomcat

//拷贝 `cp -p /usr/local/tomcat/tomcat8.5/bin/catalina.sh`

`/etc/init.d/tomcat`

/\*编辑文本\*/ `vim /etc/init.d/tomcat`

```
[root@izm5e3f5t4j8pxjmk6ip6jz /]# cp -p /usr/local/tomcat/tomcat8.5/bin/catalina.sh /etc/init.d/tomcat
[root@izm5e3f5t4j8pxjmk6ip6jz /]#
```

将tomcat中的/bin/catalina.sh 脚本 拷贝到init.d下

这样做是为了 方便启动和停止tomcat

```
[root@izm5e3f5t4j8pxjmk6ip6jz /]# vim /etc/init.d/tomcat
[root@izm5e3f5t4j8pxjmk6ip6jz /]#
```

编辑这个复制过来的文件

```
#!/bin/sh

# chkconfig: 123 63 37
# description: tomcat server init script
# Source Function Library
. /etc/init.d/functions

JAVA_HOME=/usr/java/jdk1.8/
CATALINA_HOME=/usr/local/tomcat/tomcat8.5

# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements; see the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License. You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# -----
# Control Script for the CATALINA Server
#
# Environment Variable Prerequisites
#
# Do not set the variables in this script. Instead put them into a script
# setenv.sh in CATALINA_BASE/bin to keep your customizations separate.
#
# CATALINA_HOME May point at your catalina "build" directory.
#
# CATALINA_BASE (Optional) Base directory for resolving dynamic portions
# of a Catalina installation. If not present, resolves to
# the same directory that CATALINA_HOME points to.
```

加上这几句话

```
root@izm5e3f5t4j8pxjmk6ip6jz /]# chmod 755 /etc/init.d/tomcat
root@izm5e3f5t4j8pxjmk6ip6jz /]# chkconfig --add tomcat
root@izm5e3f5t4j8pxjmk6ip6jz /]# chkconfig tomcat on
root@izm5e3f5t4j8pxjmk6ip6jz /]#
```

执行这三个命令

```
[root@izm5e3f5t4j8pxjmk6ip6jz /]# service tomcat start
Starting tomcat (via systemctl): [ OK ]
[root@izm5e3f5t4j8pxjmk6ip6jz /]#
```

就可以在 linux任何位置启动tomcat  
和关闭了



```
[root@izm5e3f5t4j8pxjm6ip6jz ~]# service tomcat start
Starting tomcat (via systemctl): [ OK ]
[root@izm5e3f5t4j8pxjm6ip6jz ~]#
```

就可以在 linux任何位置启动tomcat  
和关闭了







### ③ 安装 MySQL

#### 1. 官网下载压缩包

官网地址: <https://downloads.mysql.com/archives/community/>

 mysql-8.0.32-1.el7.x86_64.rpm-bundle.tar	2023/7/9 20:34	TAR 文件	992,410 KB
--	----------------	--------	------------

#### 2. 解压后选取需要的包上传 Linux

名称	修改日期	类型	大小
 mysql-community-server-8.0.32-1.el7.x86_64.rpm	2022/12/17 20:25	RPM 文件	65,664 KB
 mysql-community-libs-8.0.32-1.el7.x86_64.rpm	2022/12/17 20:24	RPM 文件	1,547 KB
 mysql-community-icu-data-files-8.0.32-1.el7.x86_64.rpm	2022/12/17 20:24	RPM 文件	2,167 KB
 mysql-community-common-8.0.32-1.el7.x86_64.rpm	2022/12/17 20:23	RPM 文件	654 KB
 mysql-community-client-plugins-8.0.32-1.el7.x86_64.rpm	2022/12/17 20:23	RPM 文件	2,573 KB
 mysql-community-client-8.0.32-1.el7.x86_64.rpm	2022/12/17 20:23	RPM 文件	16,530 KB

上传到路径: /usr/local/mysql

#### 3. 按顺序下载 (rpm)

```
cd /usr/local/mysql
ls -l

rpm -ivh mysql-community-common-8.0.32-1.el7.x86_64.rpm
rpm -ivh mysql-community-client-plugins-8.0.32-1.el7.x86_64.rpm
rpm -ivh mysql-community-libs-8.0.32-1.el7.x86_64.rpm
rpm -ivh mysql-community-client-8.0.32-1.el7.x86_64.rpm
rpm -ivh mysql-community-icu-data-files-8.0.32-1.el7.x86_64.rpm
yum install net-tools #安装 mysql-community-server 前需要安装好的组件
rpm -ivh mysql-community-server-8.0.32-1.el7.x86_64.rpm
```

#### 4. 启动 mysql

```
systemctl status mysqld//查看状态
systemctl start mysqld//启动
systemctl enable mysqld//开机自启
netstat -tunlp | grep mysql
# 查看 Linux 中已经启动的服务
netstat -tunlp
ps -ef | grep mysql//查看进程
```

#### 5. 登录 mysql

```
cat /var/log/mysqld.log | grep password
得到的密码记得复制
```

```

# 登录 mysql（回车后输入密码）【这里使用上面查询到的临时密码】
mysql -uroot -p
# 修改密码为 123456
SHOW VARIABLES LIKE 'validate_password%'; #查看密码安全设置
set global validate_password_policy=LOW; #设置密码安全等级为低
set global validate_password_length=4; #设置密码至少长度为 4
ALTER USER 'root'@'localhost' IDENTIFIED BY '123456';
#开启访问权限：
# 创建用户 root，密码 123456
create USER 'root'@'%' IDENTIFIED BY '123456';
flush privileges; #刷新
#给用户授权
grant all privileges on *.* to 'root'@'%';
flush privileges; #刷新
# 这里就需要修改 root 密码，如果不修改，密码就会被改为 password
ALTER USER 'root'@'localhost' IDENTIFIED BY '123456' PASSWORD EXPIRE NEVER;
# 密码与上一个命令保持一致
ALTER USER 'root'@'localhost' IDENTIFIED with mysql_native_password BY
'123456';
flush privileges; #刷新
exit;
# 开启防火墙
systemctl start firewalld
# 开放 3306 端口
firewall-cmd --add-port=3306/tcp --permanent
# 刷新添加的端口
firewall-cmd --reload

```

### 4.2.3 安全组设置

打开阿里云实例下方安全组-管理规则，设置安全组：



网卡类型：

内网

规则方向：

入方向

授权策略：

允许

协议类型：

自定义 TCP

\* 端口范围：

8080/8080

优先级：

1

授权类型：

IPv4地址段访问

\* 授权对象：

0.0.0.0/0

描述：

8080

长度为2-256个字符，不能以http://或https://开头。

确定

取消

https://blog.cdn.net/an

授权策略	优先级 ①	协议类型	端口范围 ①	授权对象 ②	描述	创建时间	操作
<div>允许</div>	1	自定义 TCP	目的: 3306/3306	源: 221.4.34.206	MySQL	2024年12月4日 21:42:39	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 5005/5005	源: 所有IPv4(0.0.0.0/0)		2024年12月3日 23:45:47	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 33060/33060	源: 所有IPv4(0.0.0.0/0)	33030	2024年12月3日 15:41:31	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 1099/1099	源: 所有IPv4(0.0.0.0/0)		2024年12月3日 00:49:00	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 80/80	源: 所有IPv4(0.0.0.0/0)	TomCat	2024年12月2日 19:13:35	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 8080/8080	源: 所有IPv4(0.0.0.0/0)	TomCat	2024年12月2日 19:13:25	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 443/443	源: 所有IPv4(0.0.0.0/0)		2024年12月2日 15:55:13	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 20/21	源: 所有IPv4(0.0.0.0/0)		2024年12月2日 15:07:56	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	1	自定义 TCP	目的: 1024/65535	源: 所有IPv4(0.0.0.0/0)		2024年12月2日 15:07:40	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	100	自定义 TCP	目的: 22/22	源: 所有IPv4(0.0.0.0/0)	System created rule.	2024年12月2日 14:41:37	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	100	全部 ICMP(IPv4)	源: -1/-1 目的: -1/-1	源: 所有IPv4(0.0.0.0/0) 目的:	System created rule.	2024年12月2日 14:41:37	<div>编辑</div> <div>复制</div> <div>删除</div>
<div>允许</div>	100	自定义 TCP	目的: 3389/3389	源: 所有IPv4(0.0.0.0/0)	System created rule.	2024年12月2日 14:41:37	<div>编辑</div> <div>复制</div> <div>删除</div>

删除

注意：3306 端口连接数据库，请勿开放到各个 ip，否则会被黑

//停止防火墙 `systemctl stop firewalld.service`  
//开启防火墙 `systemctl start firewalld.service`  
//查看默认防火墙状态  
(关闭后显示 not running 为红色字体，开启后显示 running 为白色字体)  
`firewall-cmd --state`  
//使防火墙的 80 端口开放，允许公网访问  
`firewall-cmd --zone=public --add-port=80/tcp --permanent`

[root@izmse3f5t4j8pxjmk6ip6jz ~]# systemctl start firewalld.service

[root@izmse3f5t4j8pxjmk6ip6jz ~]# firewall-cmd --state

running

[root@izmse3f5t4j8pxjmk6ip6jz ~]# systemctl stop firewalld.service

[root@izmse3f5t4j8pxjmk6ip6jz ~]# firewall-cmd --state

not running

[root@izmse3f5t4j8pxjmk6ip6jz ~]#

关闭防火墙

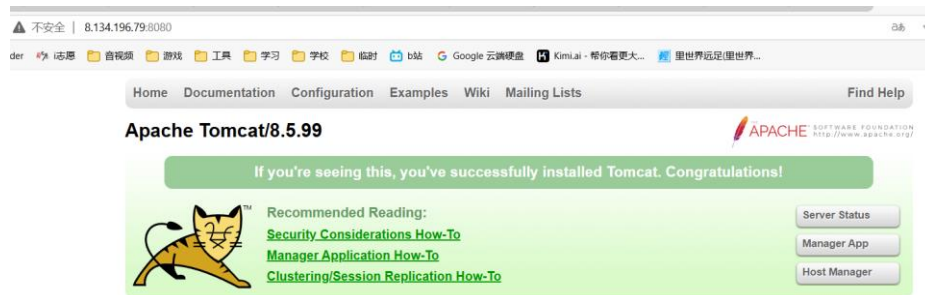


```
[root@izm5e3f5t4j8pxjm6ip6jz /]# service tomcat start
Starting tomcat (via systemctl):
[root@izm5e3f5t4j8pxjm6ip6jz /]#
```

就可以在 linux任何位置启动tomcat  
和关闭了

### 4.3 应用部署到云端

#### 1. 开启 linux 上的 tomcat 服务器

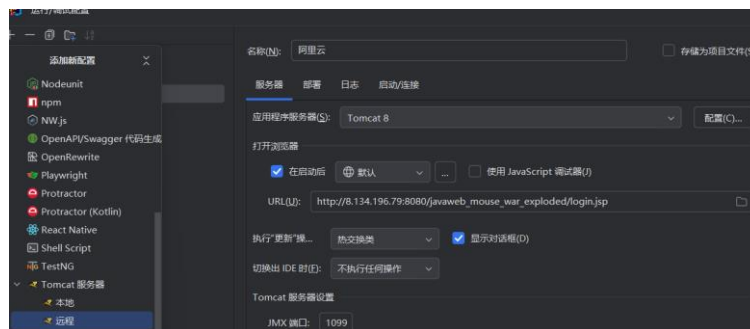


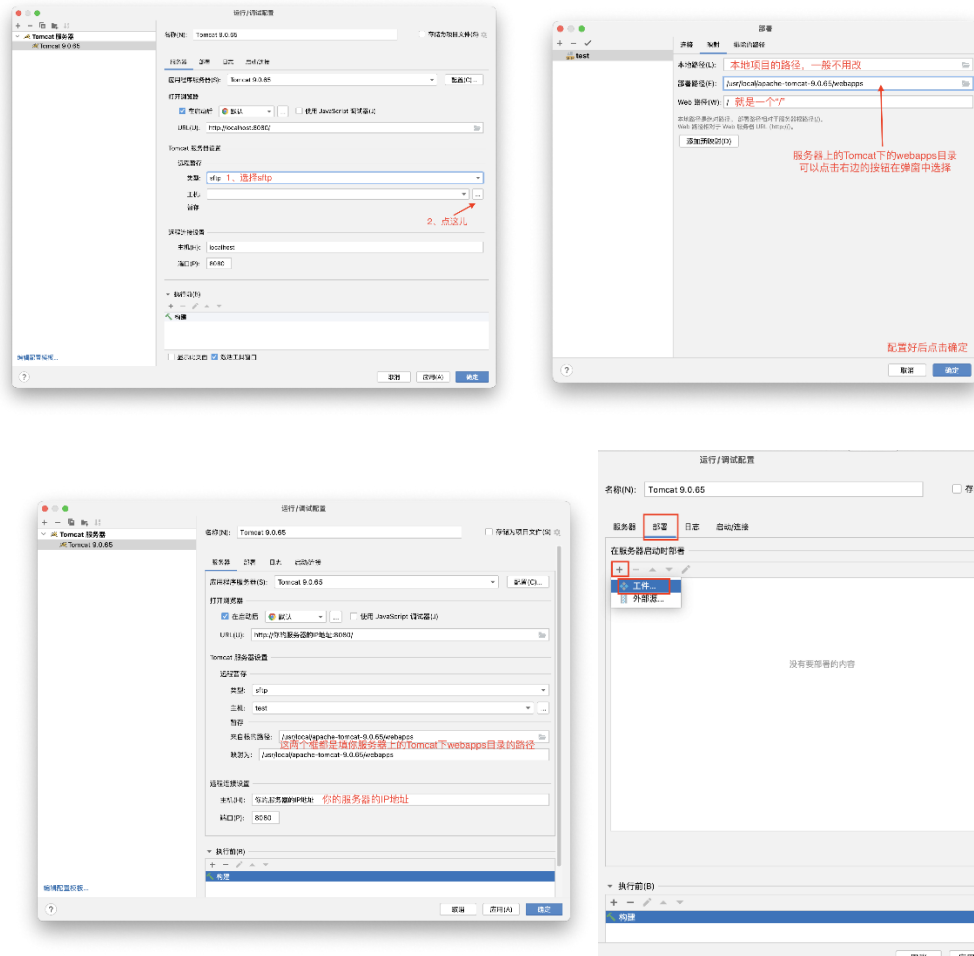
#### 2. 配置 tomcat

```
# 进入 tomcat 安装路径中的 bin 文件夹下
cd /usr/local/apache-tomcat-9.0.65/bin
# vim 打开 catalina.sh 进行编辑
vim ./catalina.sh

在 334 行左右有个# Execute The Requested Command, 加入
CATALINA_OPTS="-Dcom.sun.management.jmxremote -
Dcom.sun.management.jmxremote.port=1099 -
Dcom.sun.management.jmxremote.rmi.port=1099 -
Dcom.sun.management.jmxremote.ssl=false -
Dcom.sun.management.jmxremote.authenticate=false -
Djava.rmi.server.hostname=你的服务器的 IP 地址"
export CATALINA_OPTS
```

#### 3. 配置远程 tomcat (idea)





点击部署，即可部署成功：



## 5. 相关问题解决

### ① MySQL 的大小写区分

Linux 版本下的 MySQL 区分大小写，导致在可运行的 SQL 语句无效。这需要我们严格根据数据库中属性区分大小写重新构造 SQL 语句

### ② 数据库被黑

数据库出现 RECOVER\_YOUR\_DATA，原本数据库消失即为被黑，可以尝试重置，设置一个强的 root 密码，重置权限（除了 root 其他用户没有权限），将 root 作为本地可访问，设置个别的 ip 能连接的用户，用该用户进行登录。同时记得关闭 3306 端口的安全组，设置为只对指定 ip（自己的 ip）开放。

### ③ 远程部署后不更新

可以考虑在本地更新的 target/out 的 webapp 直接复制到远程服务器的 webapp 对应项目文件中（通过 FileZilla）

### ④ 前面的部署出现问题

如果在前面的部署过程中出现问题，可能是因为教程比较简陋，缺少了部分文件的设置，或者是因为版本不同出现问题。可以搜索查看相关问题的解决方案。

## 6. 参考文献

[在阿里云服务器上部署 Tomcat 详细图文详解-阿里云开发者社区⑩① 详解 Linux 安装 MySQL 8.0【保姆级教程】-阿里云开发者社区](#)  
[Maven 远程部署 tomcat-阿里云开发者社区](#)  
[最详细完整，使用 idea 远程部署 Tomcat（包括一些注意事项和不容易注意到的坑）\\_idea 部署远程 tomcat-CSDN 博客](#)

## 小结

通过设计一个网站从搭建到本地测试、完善、部署的整个过程，我进一步加深了对于网络应用开发的认识。这个任务对曾经不熟悉 java 与网络编程的我来说难度颇高，在整个开发的过程中我学会了网络编程 servlet 与 jsp、js、css 等技术的许多基础知识，问题调试 debug 的过程中也是让我对 servlet 与数据库有了进一步的了解，在这个过程中我的数据库遭到攻击，成功恢复后让我认识到了网络安全防范的重要新，而云服务器器的环境搭建让我对 linux 系统和阿里云服务器有了更全面的认识，操作变熟悉了。得益于人工智能技术的发展，同学们的互相帮助和老师的指导，我最终如期完成了任务。

## 指导教师评语及成绩

评语：

成绩：

指导教师签名：

批阅日期：