|  |
| --- |
| **《Web应用开发技术》**  **实验报告**  项目名称 Web应用开发技术实验  专业班级 软件工程2004 2005班  学 号  姓 名 马啸塽,李均浩,周俊林  **实验成绩：**  **批阅教师：宋铁**  2022年 1 月 12 日 |

⚡Power-Pets⚡

# 小组成员分工

**马啸塽：实验一中完成了商品的搜索功能，购物车管理及购物车持久化的功 能；在实验二中完成了查询商品的自动补全功能，并且完成了鼠标移动到商品上时的悬浮窗功能。**

**周俊林：实验一中制作了登录界面的这一块设计，包含了验证码的设计，并且在注册新用户的时候还使用了Ajax的异步请求功能，在不需要更新整个页面的情况下便可以对用户的注册的信息进行判断，通过对数据库的访问，可以知道用户信息是否已经存在，并且告述注册的用户此用户名是否已经使用。**

**李均浩：实验一中完成了用户注册模块，部分UI的重新编写，部分Java Bean 的编写，部分业务逻辑的编写，商品品类模块，用户信息设置模块；实验二中完成了UI的建造，主界面的重构，地址模板簿功能及其遮罩窗口模块，Ajax更新用户信息设置，Ajax更新购物车数据。**

# 项目完成内容

**一、*Ajax*技术应用模块**

1. **在任意页面的顶栏搜索框中运用Ajax技术完成了自动补全功能**

**理由**：可以在用户搜索宠物时向服务器发送请求，将用户正在搜索的keywords有关的所有商 品显示出来，降低了用户搜索的难度，并且该请求不需要客户端一直对服务器端的回应进行等 待。

**具体实现方法**：在客户端用JavaScript向服务器端发送Ajax请求，并指定回调函数完成搜索的自动补全。

**核心代码**

//前端发送请求部分function search(word) {

// console.log(word); var searchText = word;

xhr = new XMLHttpRequest(); xhr.onreadystatechange = process;

xhr.open("GET", "main?action=searchAction&searchText=" + searchText,

true);

xhr.send(null);

}

//服务端处理

if (req.getParameter("action").equals("searchAction")) {

String keywords = req.getParameter("searchText");

PetDaoImpl petDaoImpl = new PetDaoImpl();

StringBuilder petList = new StringBuilder();//将要返回的结果

List<Pet> petListSelectByProjectName = petDaoImpl.vagueSelectByProjectName(keywords);

List<Pet> petListSelectByName = petDaoImpl.vagueSelectByName(keywords);

boolean notHaveLetter = true;

//先判断是否为纯数字序列

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

if (keywords.charAt(i) > '9' || keywords.charAt(i) < '0') { notHaveLetter = false;

break;

}

}

//如果是id

if (notHaveLetter) {

if (petDaoImpl.selectById(Integer.parseInt(keywords)) != null) {

petList.append(petDaoImpl.selectById(Integer.parseInt(keywords)).getPetName()). append("\*");

}

}

//如果是名字

if (petListSelectByName.size() > 0) {

for (int i = 0; i <= petListSelectByName.size() - 1; i++) {

// petList.add(petListSelectByName.get(i));

petList.append(petListSelectByName.get(i).getPetName()).append("\*");

}

}

//如果是品种

if (petListSelectByProjectName.size() > 0)

for (int i = 0; i <= petListSelectByProjectName.size() - 1; i++)

{

// petList.add(petListSelectByProjectName.get(i));

petList.append(petListSelectByProjectName.get(i).getProduct()).append("\*");

}

resp.setContentType("text/plain"); PrintWriter out = resp.getWriter();

// out.println(petList.get(0)); out.print(petList);

out.flush();

out.close();

}

function process() {

// console.log("113");

if (xhr.readyState == 4) { if (xhr.status == 200) {

// console.log("114");

var $deleteli = $('li');

$deleteli.remove();

var msg = xhr.responseText;

// console.log("msg" + msg);

// console.log(msg.length); if (msg.length < 1) {

// console.log("xiaoyu1")

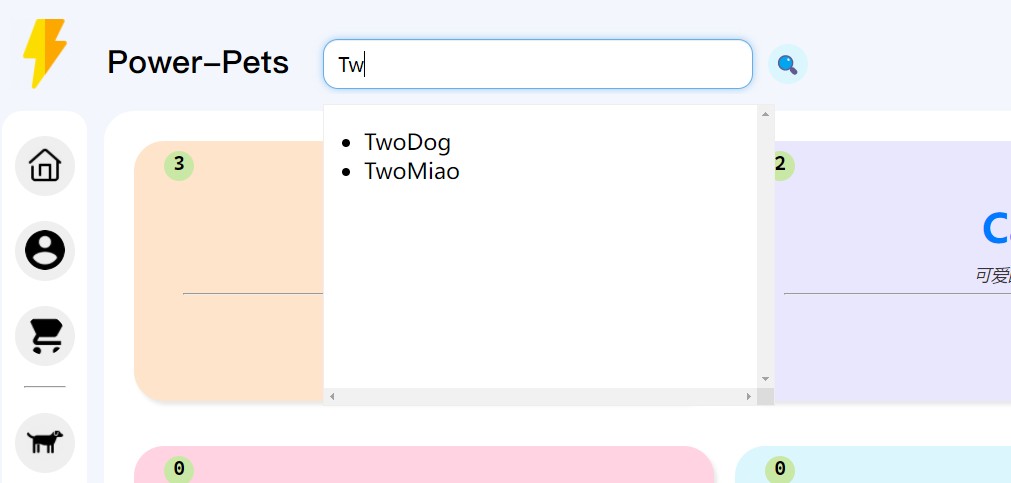
var $newul = $('<li>没有这样的宠物呢</li>');

$('ul').append($newul);

} else {

// console.log("dayuyi") var pro = "";

//以上截取部分

**最终效果图：**

1. **依靠Ajax更新、请求数据的持久化地址簿模块**

**理由**：方便用户在下单的时候无需页面跳转，即可使用以往保存设置过的收货信息，亦可以立即新 增地址。

**具体实现方法**：使用数据库对这一信息进行存储，实现数据持久化。用户查看地址簿时，向服务器 发送一个Ajax请求，服务器查询该用户的地址簿信息并发送给前端，前端接收到数据之后进行展 示；用户更改地址信息后也会向服务器发送一个Ajax请求，服务端将会对数据库进行更新。

**核心代码**

//servlet部分

if (req.getParameter("action").equals("get-address-book")) { UserDaoImpl userDaoImpl = new UserDaoImpl(); String addressBookStr =

userDaoImpl.selectById(user.getId()).getAddress();

addressBookStr = java.net.URLEncoder.encode(addressBookStr, "UTF-

8");

resp.setContentType("text/plain"); PrintWriter writer = resp.getWriter(); writer.print(addressBookStr);

}

//Ajax

$('#address-book-btn').click(function () {

$.ajax({

type: "GET",

url: "account?action=get-address-book", success: function (data) {

data = decodeURIComponent(data);

if (data === 'loginCertificateNeeded') { window.location.href = "login";

} else {

//此函数为选项卡的div设置具体数据setupDisplayTabs(data); isInitData = true;

$('.mask').removeClass('hide').fadeIn();

}

},

error: function (data) { console.log(data);

$('.prompt\_text').text('数据请求失败，请稍后再试 ~');

$('.mask').removeClass('hide');

}

});

})

**最终效果图**：

**（可在账户设置页面调出）**



**（可在下订页面调出）**



**（遮罩弹窗）**



1. **使用Ajax来更新购物车数据，而不需要刷新整个页面**

**理由**：如果不使用Ajax，就无法部分刷新网页，异步更新数据。如果要更新类似于物品数量这样的数据，就会产生很多不便。

**具体实现方法**：当购物车的数据发生变化，那么将会用Ajax技术向服务器发送变化的购物车数据， 以此来更新信息。

**核心代码**

//前端部分

$('#cart-table tr').each(function (i) {// 遍历 tr let unit\_price = 0;

let amount = 0;

let item\_id = ($(this).attr("id"));

current\_tr = $(this);

$(this).children('td').each(function (j) { //遍历 tr 的各个 td if (j === 2) {

unit\_price = (Number)($(this).text());

}

if (j === 3) {

let input\_amount = $(this).children().children().children(".col- md-6").children().children();

amount = (Number)(input\_amount.val());

input\_amount.change(function (e) { if ($(this).val() < 0) {

$(this).val(0);

}

if ($(this).val > 100) {

$(this).val(100);

}

refreshTotalPrice();

})

let btn\_add = $(this).children().children().children(".div- add").children();

btn\_add.click(function (e) { if (amount >= 100)

return; amount++;

input\_amount.val(amount); refreshTotalPrice();

})

let btn\_minus = $(this).children().children().children(".div- minus").children();

btn\_minus.click(function (e) { if (amount <= 0)

return; amount--;

input\_amount.val(amount); refreshTotalPrice();

})

}

if (j === 4) {

$(this).text(((Number)(unit\_price \* amount)).toFixed(2));

}

if (j === 5) {

$(this).children("button").click(function (e) {

$(this).parent().parent().remove();

$('.wrapper').height(Math.max((($("#cart-table tr").length)

\* 110 + 450), 1080));

// console.log("deleteCartItem is called");

$.ajax({

type: "GET",

url: "cart?action=delete-from-cart&petID=" + item\_id, success: function (message) {

// console.log(message);

if (message > 0) {

console.log("成功更新了购物车数据：" + message);

}

},

error: function (message) {

console.log("更新购物车数据时出错：" + message);

}

})

});

}

});

});

//服务端部分

if (action.equals("update")) {

String newCartData = req.getParameter("data"); String[] petsString = newCartData.split("#"); int count = 0;

for (String s : petsString) {

count += Integer.parseInt((s.split("@"))[1]);

}

CartDaoImpl cartdaoimpl = new CartDaoImpl(); PetDaoImpl petdaoimpl = new PetDaoImpl(); List<Pet> cartList = new ArrayList<>();

Cart nowCart = new Cart(Integer.parseInt(userID), newCartData,

count);

cartdaoimpl.updateCart(nowCart); String petString = nowCart.getPetsID(); if (!petString.equals("")) {

String[] petsIdAndAmount = petString.split("#"); for (String petIDAndAmount : petsIdAndAmount) {

int amount = Integer.parseInt((petIDAndAmount.split("@"))

[1]);

int petID = Integer.parseInt(((petIDAndAmount.split("@"))

[0]));

Pet newPet = petdaoimpl.selectById(petID); newPet.setAmount(amount); cartList.add(newPet);

}

}

req.getSession().setAttribute("cartList", cartList); req.getSession().setAttribute("cartAmount", nowCart.getAmount());

resp.setContentType("text/plain"); PrintWriter writer = resp.getWriter(); writer.print("UPDATE SUCCESS"); writer.flush();

writer.close();

}

**最终效果图**



## Ajax验证用户名是否可用

**核心代码**

<%

VerifyCodeUtil.Builder builder = new VerifyCodeUtil.Builder(); VerifyCodeUtil verifyUtil = builder.build();

Object[] o = verifyUtil.createImage(); String verifyCode = (String) o[0];

BufferedImage verifyCodeImage = (BufferedImage) o[1];

System.out.println(verifyCode); session.setAttribute("verifyCode",verifyCode);

int size = 200;

Graphics2D g = verifyCodeImage.createGraphics(); g.setRenderingHint(

RenderingHints.KEY\_ANTIALIASING, RenderingHints.VALUE\_ANTIALIAS\_ON);

g.setColor(Color.BLUE);

for (int i = 0; i < size; i += 5) { g.drawOval(i, i, size - i, size - i);

}

g.dispose();

ByteArrayOutputStream baos = new ByteArrayOutputStream(); ImageIO.write(verifyCodeImage, "png", baos);

String data = DatatypeConverter.printBase64Binary(baos.toByteArray()); String imageString = "data:image/png;base64," + data;

String html = "<div class=\"col-md-4 column\"><img src='" + imageString + "'></div></div></div>";

out.println(html);

%>

var xhr;

function checkUsername(){

var username= document.getElementById("input\_username").value; xhr=new XMLHttpRequest();

xhr.onreadystatechange=fun1; xhr.open("get","UsernameExit?username="+username,true);

xhr.send(null);

}

function fun1()

{

if(xhr.status===200) if(xhr.readyState===4)

{

var responseInfo=xhr.responseText;

document.getElementById("usernametips").innerText=responseInfo;

}

}

**最终效果图**





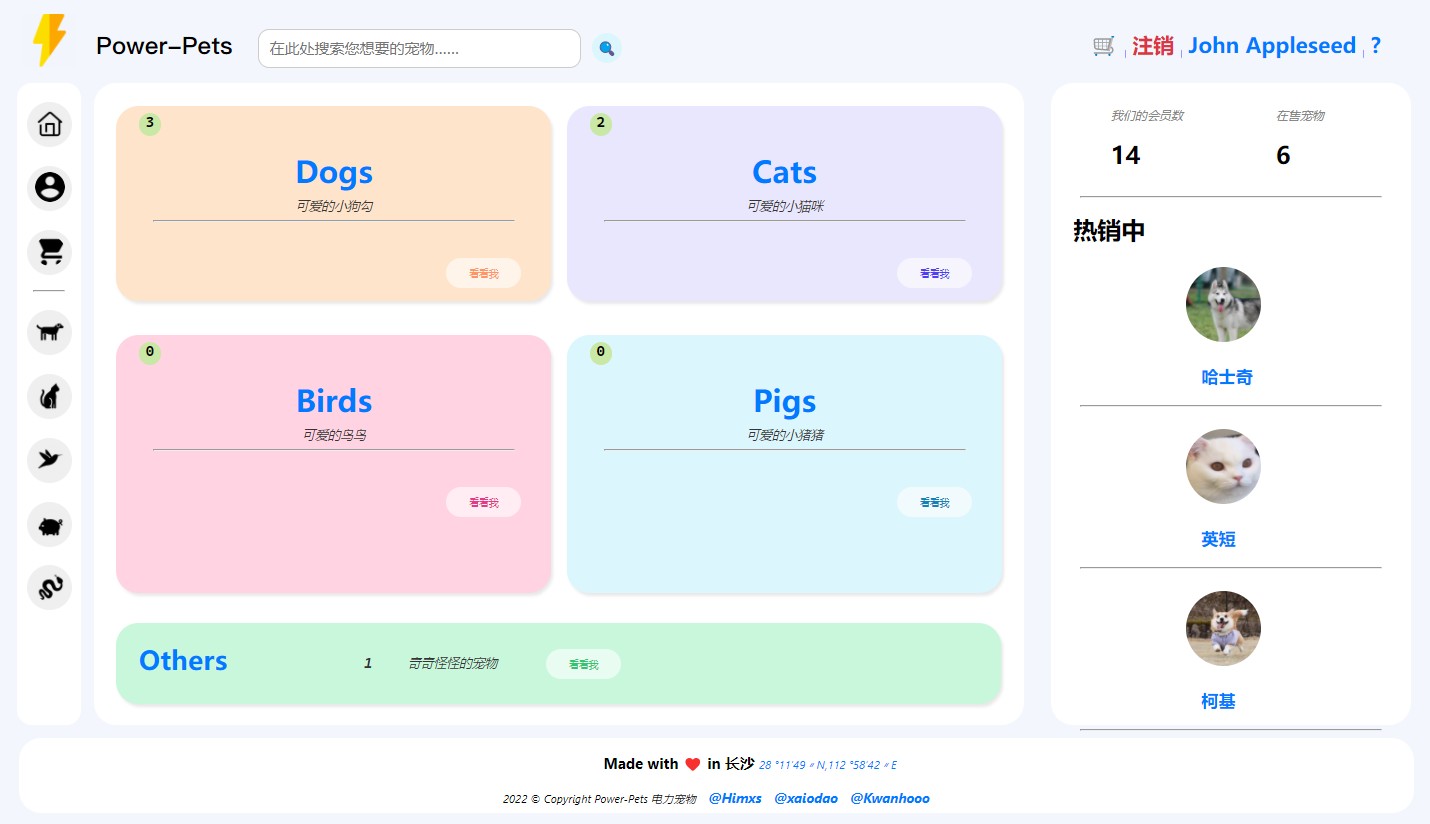
1. **使用Ajax修改了其余前后端交互部分**

如用户信息设置模块的表单信息更新、提交

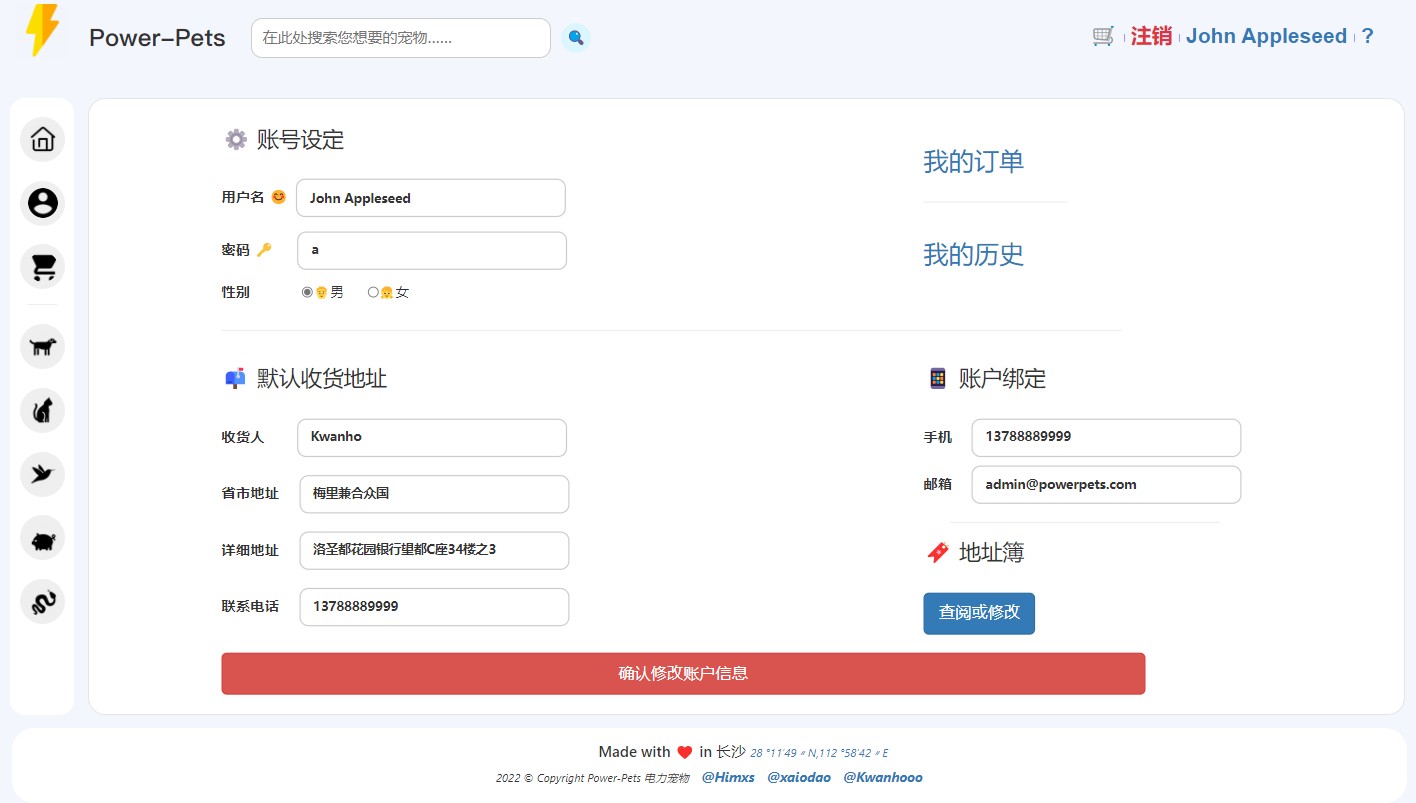
## 二、用户界面建造

**主界面**

主界面使用原生HTML以及CSS编写，主要由顶栏（用于展示标题logo、搜索、跳转至账户相关的 页面）；中部的主框，主框包含左侧导航栏（方便快速跳转至各个门类），中部的门类仪表盘，右 侧的统计、宠物推荐栏；以及底栏（用于展示一些网站信息）。



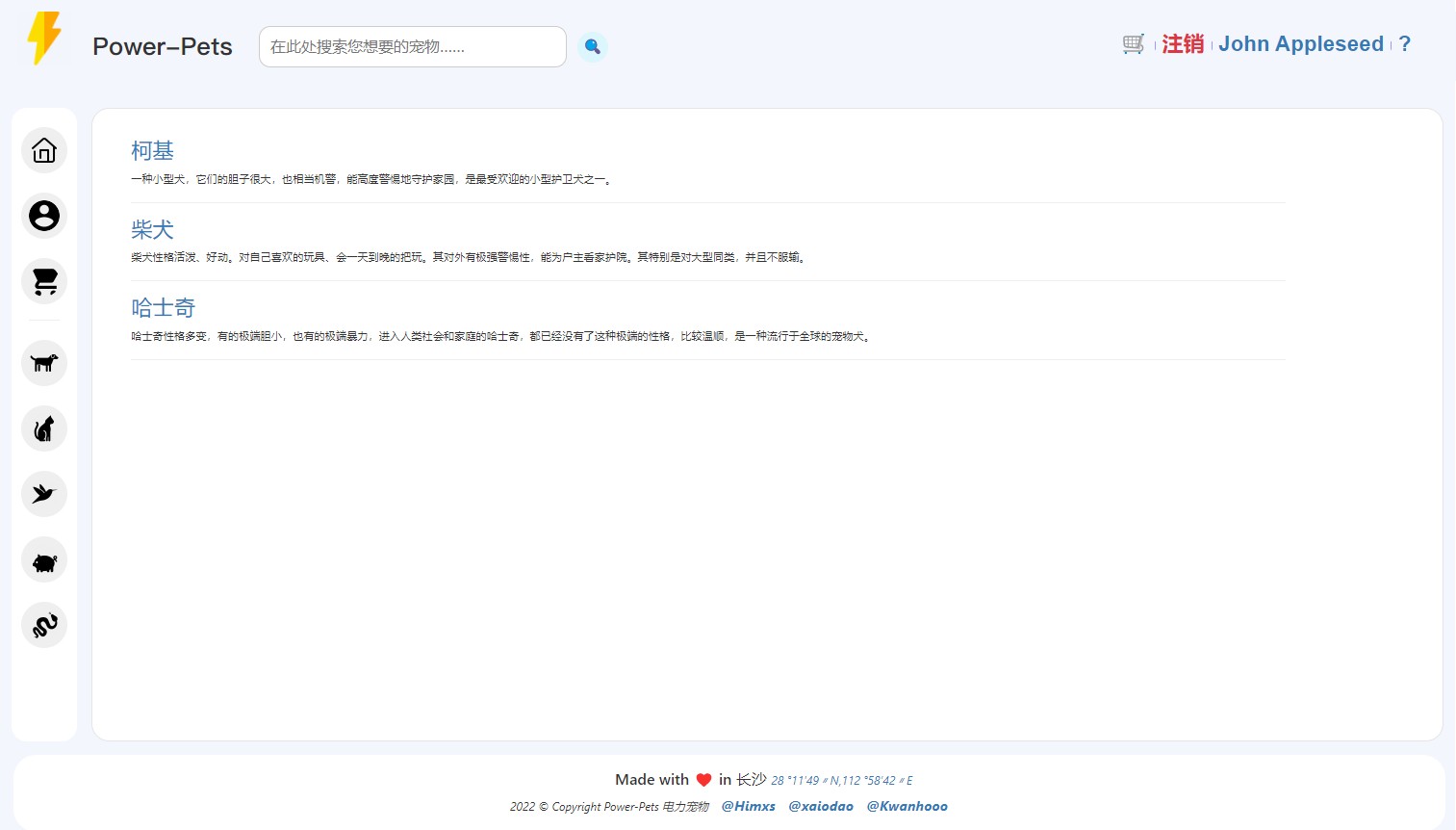
**账户设置界面**

此界面提供了一些分类布局过的输入框，允许用户查看、更改账户信息。

**购物车界面**

用户可以在此界面管理购物车，进行下单操作。![image-20220113223015872] (https://s2.loli.net/2022/01/14/ygdoNQbC9qUEaxP.png)

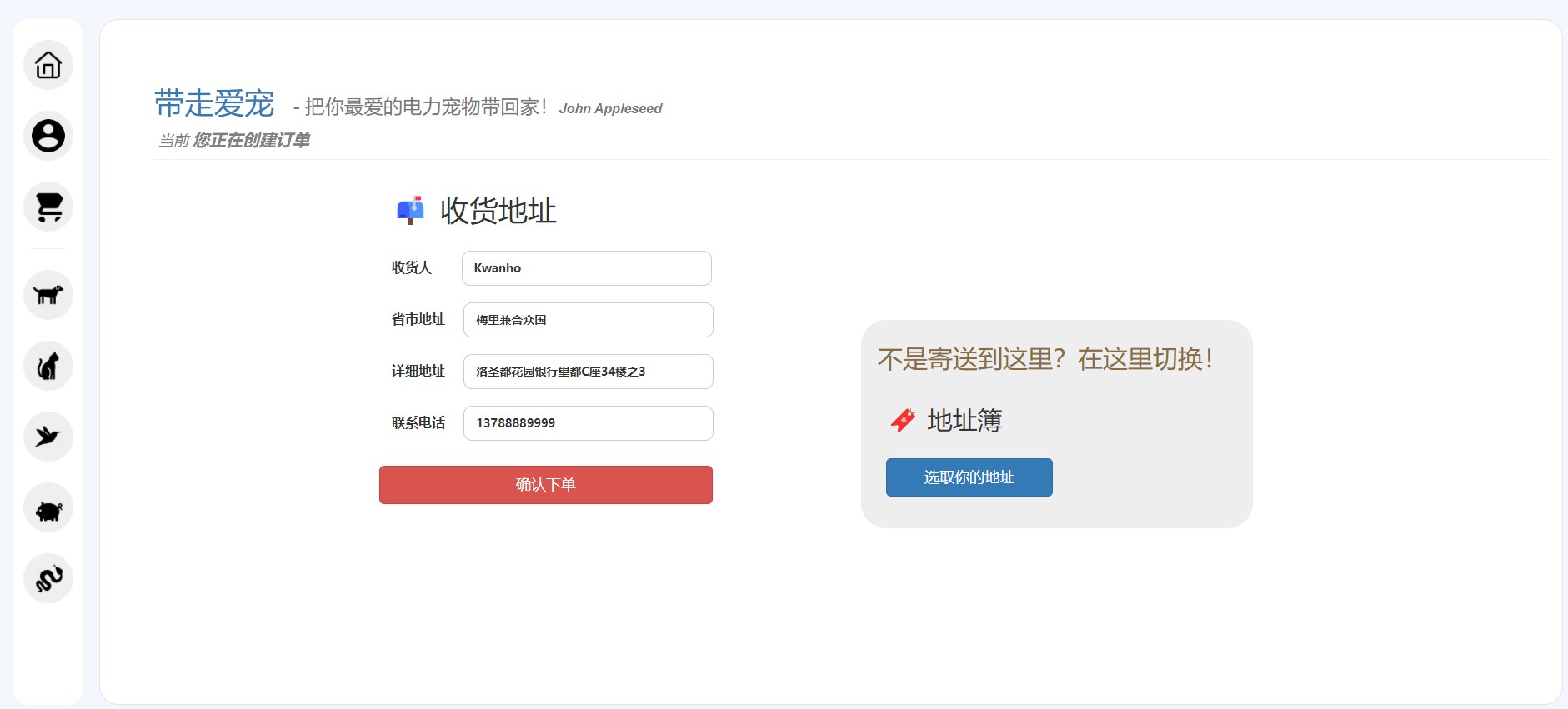
**品类展示界面**



**商品介绍页面**



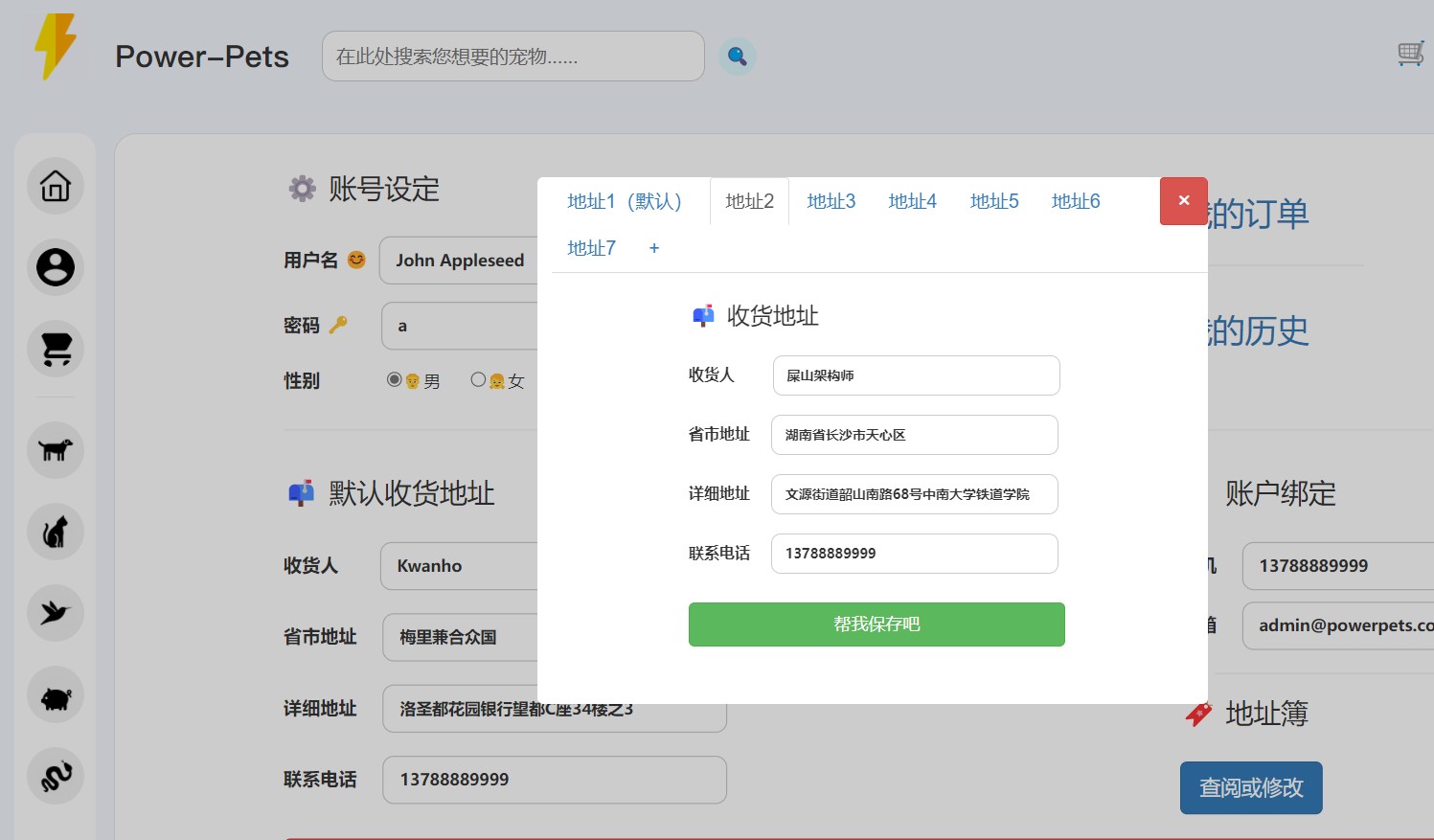
**下单界面**



**订单生成界面**



**地址簿界面**



# 体验本项目

**请跳转至：*http://120.78.235.195:8080/PowerPets/***