智能宠物追踪

综合平台

项目设计书

项目名称 智能宠物追踪综合平台

年级专业 16软件工程

组 长 林志舟

小组成员 林志舟085 郭姝然044

2019年6月

目　　录

[第一章 引言 3](#_Toc10662081)

[1.1 编写目的 3](#_Toc10662082)

[1.2 市场背景 3](#_Toc10662083)

[1.3 定位技术的选择 4](#_Toc10662084)

[1.4 术语定义及说明 4](#_Toc10662085)

[1.5 状态检测工具 9](#_Toc10662086)

[第二章 设计概述 11](#_Toc10662087)

[2.1 任务和目标 11](#_Toc10662088)

[2.2 需求概述 12](#_Toc10662089)

[2.2.1 移动端 12](#_Toc10662090)

[2.2.2 Web端 12](#_Toc10662091)

[2.3 开发环境与工具 12](#_Toc10662092)

[2.3.1 移动端 12](#_Toc10662093)

[2.3.2 服务端 12](#_Toc10662094)

[2.4 限制条件 13](#_Toc10662095)

[2.5 模块概述 13](#_Toc10662096)

[第三章 需求分析与可行性分析 14](#_Toc10662097)

[3.1 接口分析 14](#_Toc10662098)

[3.2 技术可行性 15](#_Toc10662099)

[3.2.1 导航实现基本逻辑 15](#_Toc10662100)

[3.2.2 宠物经纬度获取 15](#_Toc10662101)

[3.2.3 网络通信问题 15](#_Toc10662102)

[3.2.4 经济可行性 16](#_Toc10662103)

[3.2.5 社会环境分析 16](#_Toc10662104)

[3.2.6 应用领域分析与拓展 16](#_Toc10662105)

[3.2.7 结论 16](#_Toc10662106)

[第四章 总体方案确认 17](#_Toc10662107)

[4.1 系统总体结构确认 17](#_Toc10662108)

[第五章 系统详细设计 18](#_Toc10662109)

[5.1 系统结构设计及子系统划分 18](#_Toc10662110)

[5.2 系统功能模块详细设计 18](#_Toc10662111)

[5.3 硬件平台介绍 19](#_Toc10662112)

[5.3.1 硬件开发大致流程 19](#_Toc10662113)

[5.3.2 通信问题的解决 20](#_Toc10662114)

[5.4 代码解释及接口信息 21](#_Toc10662115)

[5.4.1 接口具体代码 21](#_Toc10662116)

[5.4.2 导航介绍 24](#_Toc10662117)

[5.4.3 服务端代码介绍 26](#_Toc10662118)

[5.4.4 客户端与服务端交互部分代码 30](#_Toc10662119)

[第六章 完整功能介绍 34](#_Toc10662120)

[6.1 网站功能 34](#_Toc10662121)

[6.1.1 已完善功能展示 34](#_Toc10662122)

[6.1.2 错误拦截界面 37](#_Toc10662123)

[6.2 移动端功能 38](#_Toc10662124)

[6.2.1 宠物导航模块 39](#_Toc10662125)

[6.2.2 宠物管理模块 42](#_Toc10662126)

[6.2.3 用户管理模块 43](#_Toc10662127)

[第七章 系统测试与部署 46](#_Toc10662128)

[7.1 测试 46](#_Toc10662129)

[7.2 权限配置 46](#_Toc10662130)

[7.3 完成情况 47](#_Toc10662131)

[7.4 困难总结 47](#_Toc10662132)

[7.4.1 接口版本问题 47](#_Toc10662133)

[7.4.2 虚拟机上的导航功能问题 48](#_Toc10662134)

[7.4.3 调用外部库问题 48](#_Toc10662135)

[7.5 经验总结 48](#_Toc10662136)

[参考文献 49](#_Toc10662141)

# 引言

## 编写目的

宠物行业是社会经济发展进入较高水平时衍生出的新兴产业，未来,随着宠物饲养观念的更广泛普及和宠物行业延伸服务的更深层挖掘，我国宠物行业的市场空间将进一步扩大。未来3-5年宠物行业将继续维持20%以上的高增长，预计2018年中国宠物行业规模将达到1678亿元，到2022年市场规模有望突破2500亿元[1]。随着宠物行业市场空间的进一步扩大，宠物的基数增大，丢失宠物的的基数也在增大，用户寻找丢失宠物的需求也更为迫切。

综上所述，为了适应逐渐增长的需求，同时也为了更加科学、高效地开发出一款具有智能追踪宠物功能的app，编写此文档，同时将此文档作为该项目其他文档的标杆。

## 市场背景

随着我国宠物产业进入蓬勃发展期。人均GDP的提升、宠物饲养比例的提高、宠物消费意愿的增强等因素，宠物产业持续发展。并且，宠物数量仍然具有很大的提升空间，量变阶段远未结束。

宠物行业市场的进一步扩大以及人们生活条件的改善，使得越来越多的人开始饲养宠物，宠物走丢发生的频数也越来越大，由此，寻找丢失宠物的需求应运而生。同时，老龄化的社会也带来了宠物需求的兴起，为了减少老年人因为宠物丢失而带来的更加沉重的落寞，减少老年人因身心不愉悦而诱发的身体疾病，同时减轻子女现阶段的负担等，开发出这样一款集成Web端、移动端的智能宠物追踪平台。

## 定位技术的选择

定位技术我们并不陌生，从航海、航天、航空、测绘、军事、自然灾害预防等领域，再到日常生活中的人员搜寻、位置查找、车辆导航与线路规划等等，各种定位技术都有广泛应用 [2]。GPS和基站定位技术基本满足了用户在室外场景中对位置服务的需求。然而，我们有80%的时间是在室内度过的，个人用户、服务机器人、新型物联网设备等大量的定位需求也发生在室内；而室内场景受到建筑物的遮挡，GNSS信号快速衰减，甚至完全拒止，无法满足室内场景中导航定位的需要[3]。

和室外定位相比，室内定位面临很多独特的挑战。

但是结合该应用的使用场景，我们不对室内定位技术过多讨论。因为宠物的走丢大多发生在室外，且宠物的活跃状态也不适合采用室内定位技术。而室外的GPS定位尽管误差比室内定位要大（通常在10m左右），但是在宽阔的室外，10M已经足以帮助宠物主人发现宠物。且成本较室内定位比也更低，更容易实现。所以我们选择采用室外定位技术。

## 术语定义及说明

·GPS（Global Positioning System）: 全球定位系统的简称，全称为定时测距导航卫星全球定位系统。GPS全球全天候定位、定位精度高、观测时间短、可提供全球统一的三围地心坐标，主要功能为导航、测量以及授时[4]。其定位原理是测量出已知位置的卫星到用户接收机之间的距离，然后综合多颗卫星的数据就可以知道接收机的具体位置，且定位精度在民用领域开放为10米。

·GPRS（General Packet Radio Service）: 通用分组无线服务技术的简称，是GSM（Global System For Mobile Communications，全球移动通信系统）移动电话用户可用的一种移动数据业务，属于第二代移动通信中的数据传输技术。其应用特点为数据传输速度高、永远在线、仅按数据流量计费，并且首先引入了分组交换的传输模式，使得原本采用电路交换模式的GSM传输数据方式发生了根本性的变化，这在无限资源稀缺的情况下显得尤为重要[5]。

·AGPS（Assisted Global Positioning System）: 辅助全球卫星定位系统的简称，利用手机基地站的资讯，配合传统的GPS卫星，让定位的速度变得更快。A-GPS技术是一种结合了网络基站信息和GPS信息对移动台进行定位的技术，可以在GSM/GPRS、WCDMA和CDMA2000网络中使用[6]。该技术需要在手机内增加GPS接收机模块，并改造手机天线，同时要在移动网络上加建位置服务器、差分GPS基准站等设备。

·BT4.0（BlueTooth4.0）: 蓝牙4.0，是2012年最新蓝牙版本，是3.0的升级版本，相较于3.0的优势是更省电、距离更远、成本更低。这项技术可为制造商及用户提供三种无线连接方式，包括用于多个类别电子消费产品的传统蓝牙技术、用于手机、相机、摄像机、PC及电视等视讯、音乐及图片传输的蓝牙高速技术，以及用于保健及健康、个人设备、汽车及自动化行业的低功率传感设备和新的网络服务的蓝牙低耗能技术。基于GPRS和蓝牙技术的物联网方案可以通过GPRS将采集数据传输到数据中心，同时在没有云端连接的情况下，也可直接通过蓝牙终端进行监测管理，且不用拆卸设备连线等繁琐的操作[7]。

·MQTT（Message Queuing Telemetry Transport）协议：是一种基于发布/订阅的轻量级消息传输协议，它的出现为Android平台上实现消息推送提供了新的途径，其设计思想是开放、简单、轻量、易于实现，适合在带宽、计算和处理能力受限的环境下工作[8]。

MQTT可以分为两部分：MQTT客户端，MQTT消息代理[9]。MQTT客户端直接使用MQTT协议与MQTT消息代理相连接。每个MQTT命令消息的消息头都包含一个固定的报头。某些消息也需要包含一个可变的报头和一个payload。并且，MQTT的报头固定为2bytes，这不仅节省了开销，而且降低了网络流量的消耗[10]。作为在手机上的发展前景十分可观的一种协议，MQTT在即时通信中的应用实现也贡献了很大的作用。

具体说明为：

（1）当用户进行即时通信的消息发送时: 用户输入想要发送的文本, 然后点击发送按钮, 或者选择图片、语音按钮发送语音 或图片消息(语音和图片是通过 Socket上传到数据服务 器上的, 数据服务器返回相应的下载链接), 客户端先将消息内容传送到消息编解码模块进行编码。编码后, 通过 MqttClient.publish()方法将消息发布到即时通信的话题上, 然后由MQTT服务器进行处理并返回消息, MQTT 客户端接收消息, 由编解码模块进行解码, 如果 发布成功, 表明 MQTT 服务器已将消息转发到目标用户, 并且客户端会刷新消息列表; 否则重新发送消息。

（2）当用户进行即时通信的消息接收时: 如果用户接收到其他用户发送来的消息, 首先, MQTT客户端将接收到的消息传递给消息 编解码模块, 然后, 判断消息类型, 如果是文本消息, 直接保存到本地数据库, 并将消息显示在消息列表上; 如果是语言或图片消息, 先通过下载地址下载, 并保存在 sdcard 中, 然后再显示到消息列表上[11]。

总的来说，在网络代价昂贵，带宽低、不可靠的环境或者处理器和内存资源有限的环境下，MQTT成为了M2M、物联网以及移动应用的理想选择[12]。

·WZ-203CS开发板，是一款具有低功耗、体积小、定位块等功能的优秀开发板，采用MT2503D芯片组的智能物联网模块，通过“智云服”一站式智能硬件开发平台，快速完成智能硬件产品搭建、研发测试及二次开发。其三大性能依次为：超低休眠功耗、快速休眠唤醒和GPS北斗快速定位。203C是一款基于MT2503D芯片平台开发的GSM/GPRS/GPS/Beidou多功能模块，并采用LCC主流封装[13]。MT2503D平台是GSM平台MT6261和GNSS平台MT3333的组合体。其在面积缩小40%、功能大大增加的同时，性能也有很大程度的提高。WZ-203CS的定位功能依赖于LOCUS技术，这是一种自主日志信息记录功能，能把卫星导航数据存储到模块内部的闪存，便于用户精准分析卫星导航数据。

MT2503，用于对该系统开发板WZ-203CS的集成，它是基于高度集成的超小型系统级封装，整合蓝牙3.0、多星系GNSS系统和2G

基带，搭载集成内存的armv7微控制潜在应用领域包括具备简单应用功能的可穿戴设备、可移动的资产跟踪设备、注重安全的工业应用。

·关于开发板，资源提供为：

（1）USB 5V供电，内置串口转USB芯片，普通MicroUSB可以支持调试和烧写

（2）模块多有的接口引出，可以快速的二次开发

（3）模块资源：2路串口，1路ADC，1路SD卡接口，1路I2C

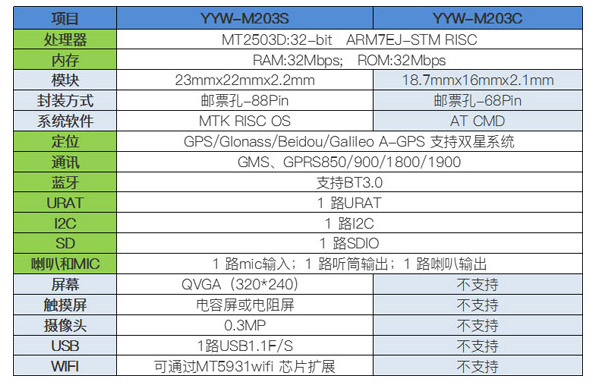
（4）超小体积开发板，客户可以直接用于产品开发

（5）LED指示灯，指示开发板工作状态

（6）GPS、GPRS、BT(4.0)天线座

（7）eSIM，nano-SIM兼容性设计

其中，开发板套件推出两种规格的MT2053模块，M203S和M203C，可以根据不通的需求选择不通模块进行开发，规格参数如图1.1所示[14]。



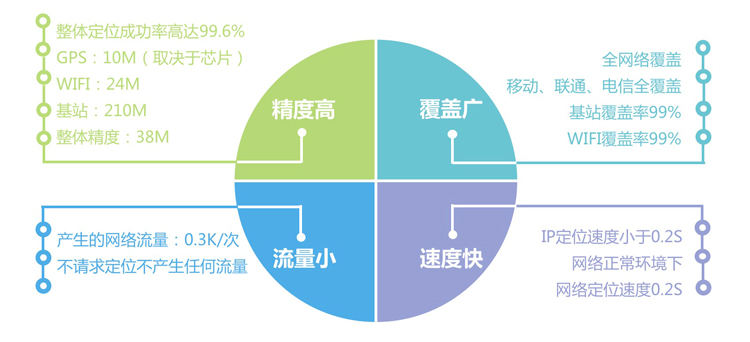


图1.1 开发板模块及特点图

## 状态检测工具

通过对开发板进行烧录，利用测试程序绑定好的开发板，能够实时获取到开发板的数据，监测开发板是否在线。根据第三方平台提供的状态检测的小程序检测硬件状态，界面如下。

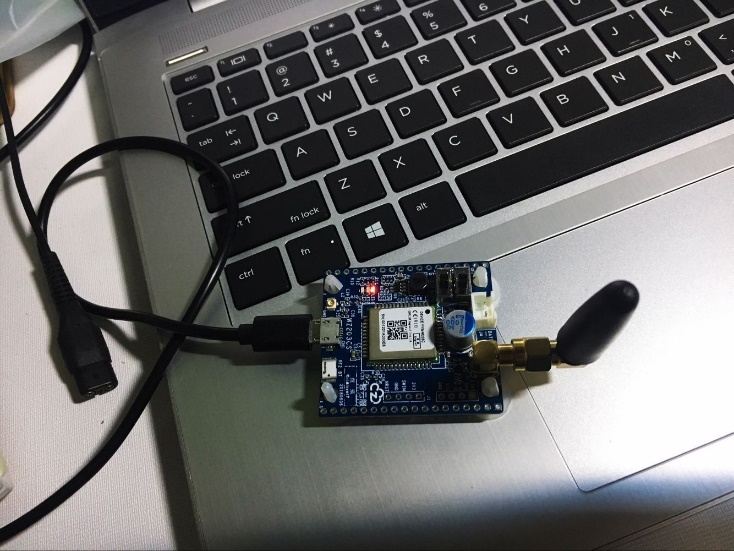


图1.2 状态检测及开发板图

# 设计概述

## 任务和目标

对于开发的分工和总体设计，我们将其分为了四个阶段，每个阶段以该阶段的文档产出作为结束的标志。流程如图2.1所示所示。

立项🡪A:设计阶段，制定开发计划，做好分工。

A🡪B:硬件设计阶段，以程序成功从硬件上取得数据作为该阶段结束的标志。

B🡪C：软件设计阶段，以产出稳定的app作为该阶段结束标志。

C🡪结束:文档整理与测试阶段，以总结报告作为该阶段结束标志。



图2.1 开发阶段图

## 需求概述

### 移动端

在该智能宠物追踪平台中，移动端的核心需求是能够对遗失宠物的位置定位并规划寻找路径。

### Web端

Web端要求能够处理移动端发出的注册登陆请求。

能够处理设备与用户之间的关系，确保用户在移动端能够实现开发板经纬度的获取。

能够对用户信息进行管理（社交信息）。

能够有可视化且美观的网页界面，可以通过域名进行访问。

有足够的信息安全保障同时响应时间的“2-5-8原则”。

## 开发环境与工具

### 移动端

移动端的开发基于Android Studio，采用API27。依靠Gradle对项目版本号，依赖库，进行管理。

### 服务端

#### Spring Boot服务端

服务端的开发基于IDEA，依靠Spring Boot+Hibernate+Maven框架。云服务端基于Ubuntu运行，在开发完成后将项目打包成jar包在阿里云上运行。该服务端主要为移动端提供服务，以及处理用户对网站的各种请求。

#### MQTT服务端

为节约开发时间与降低开发难度，采用智云服平台开发好的阿里MQTT服务端。对硬件数据进行接收及处理。

## 限制条件

·由于目前各个Wi-fi模块厂商的Smart Config协议均未完全成熟，所以5G路由器信号未能得到使用支持。

·开发小组成员的专业都是软件工程，对硬件方面的开发完成度与开发时间难以保障。

## 模块概述

项目有三大核心模块：定位模块、追踪模块、社交模块。追踪模块集成了语音模块，路径规划模块。如图2.2所示。

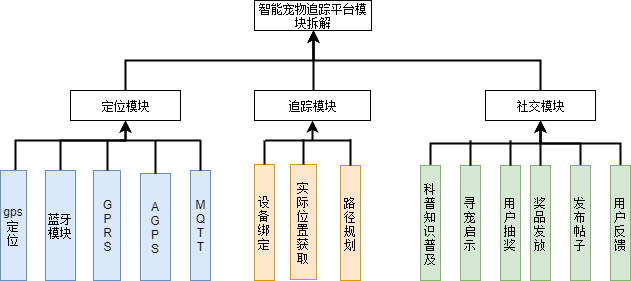


图2.2 模块拆解图

# 需求分析与可行性分析

## 接口分析

·系统接口需求分析

在Android的SDK中，SDK通过字典键值对方式进行设备控制和状态接受。当SDK接受到APP下发的指令后，对应解析为设备可识别的数据后，发送给设备。反之，当SDK收到设备或上报的数据后，对应解析为相应字典键值对后，上报给APP。如果设备定义了数据点，APP发送指令必须符合数据点的定义；如果设备没有定义数据点，设备指令可以按照透传数据以自定义格式下发。获取设备相关数据点成功后，可通过ZYFSDK发送指令来对设备进行控制，从获取的指令属性中可以判断出数据点是否可控。

·soft AP接口分析

Soft AP模式是一种通过无线网卡，使用专用软件在PC上实现AP功能的技术，它可以取代无线网络中的AP（Access Point，无线接入点），从而会降低无线组网的成本。也就是可以把载体作为无线接入点，让电脑、手机或者其他上网设备的无线网连接到载体上，然后通过载体的网络（GPRS）上网。

当设备进入SoftAP配置模式时，设备本身将成为一个AP。此时智能手机可直接与设备进行连接，然后在手机上的界面输入路由器的SSID和密码，设备接受到信息的时候会开始自动尝试连接路由器，若连接成功则自动切换到正常的模式。

## 技术可行性

### 导航实现基本逻辑

要实现导航功能，需要获取宠物的经度与纬度的精确信息以及用户自身的经度与纬度信息，再根据双方的经纬度信息调用导航接口实现导航。

### 宠物经纬度获取

通过在宠物身上佩戴一个项圈，在这个项圈中嵌入带有定位功能的芯片来实现宠物位置信息的实时获取。在团队成员充分筛选与比较后决定采用集成了MT2503芯片的WZ203CS开发板，该开发板集成了GPRS,GPS,蓝牙模块，WIFI模块等，且体积不大，可以实现宠物的定位功能以及封装后的便利性。

### 网络通信问题

现在已经拥有用户以及宠物的位置信息，并将导航功能实现，如何将宠物项圈上的位置信息发送给客户端是主要需要考虑的问题。通过查阅智云服平台的文档以及代码验证后，发现平台将芯片的位置信息映射在了一个实体类中，对该实体类进行操作即可实现对硬件设备信息的获取。

### 经济可行性

开发成功后，系统所获得的效益主要来自于两个方面，一是由用户流量所带来的相关效益，二是出售封装了芯片后的项圈所带来的效益。前期开发成本主要是云服务器租赁成本以及开发板成本。经过初步估算，在经济上可行。

### 社会环境分析

宠物经济的兴起带来的宠物基数增大，从而导致了丢失宠物数量的增多，增加了流浪宠物数量，从而带来了一些社会问题，该系统的开发，能够在一定程度上减缓该问题。且使用方便，便于推广。

### 应用领域分析与拓展

该系统的目的是进行智能宠物的追踪，但不仅限于该领域。若将芯片嵌入老人的拐杖中，可将系统拓展至寻人的范围，将芯片放入自行车电动车中，系统便可应用于寻找丢失车辆中。以及许多其他可用的、便利的且不违反法律的领域，将来只要对芯片进行相关封装，客户端无需进行更改，即可简单实现应用领域的扩展。

### 结论

通过相关调研以及对开发文档的充分查阅，可以得出结论，该系统开发在理论上可行。

# 总体方案确认

## 系统总体结构确认

按照模块分析得出，智能宠物追踪系统的总体结构可以分为三个子系统，分别是定位系统，追踪系统和社交系统。其中定位系统主要实现第三方设备的经度纬度的定位，追踪系统的主要职责是通过绑定设备来获取定位系统传过来的经度以及纬度，并将实际位置在高德地图上显示出来，然后结合手机自身的经度与纬度，进行路径规划，计算好寻找路线，预算寻找时间。社交系统的主要功能有科普知识普及，寻宠启示的发布，用户抽奖以及奖品发放，发布帖子还有接受来自用户对于软件的反馈。在智能宠物追踪系统中，支撑系统是定位子系统和追踪子系统，应用系统是社交系统。支撑系统结构图如下：



图4.1 支撑结构图

# 系统详细设计

## 系统结构设计及子系统划分

各个模块划分在第二章已经给出，这里给出各个系统下每个模块之间的调用关系。如图5.1所示。

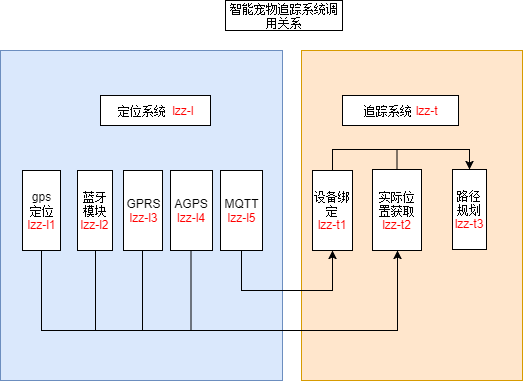


图5.1 模块调用关系图

## 系统功能模块详细设计

具体划分在第四章中已经提出，这里给出按系统划分模块的具体HIPO图。其中，定位系统的核心模块是GPS模块与GPRS模块，如图5.2所示。（请参照图5.1理解）

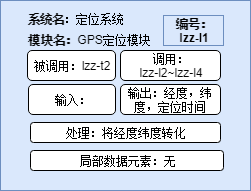
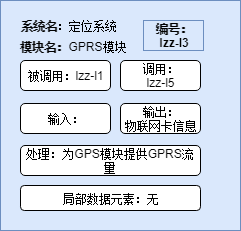
 

图5.2 GPS与GPRS模块

追踪系统的HIPO图如下：

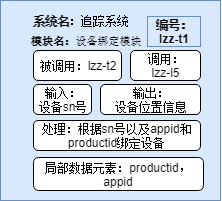
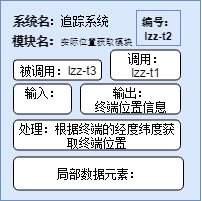
  

图5.3 追踪系统HIPO图

## 硬件平台介绍

### 硬件开发大致流程

在智云服的硬件平台选择好需要的开发板后，在云平台上选择好需要的功能后，生成烧写包以及SDK，如图5.4所示。下载烧写包，对其中的数据项进行更改，再使用串口工具进行烧录，编写程序验证开发板， SN号烧录成功即标志着硬件开发阶段的结束。流程如图5.5所示。需要注意的是图中所说的绑定指的是将设备的SN号写入设备，即设备与SN号的绑定。并非用户账号与SN号的绑定，因为该绑定需要用户的输入，所以将在软件开发阶段通过开发接口进行，不在硬件阶段进行。

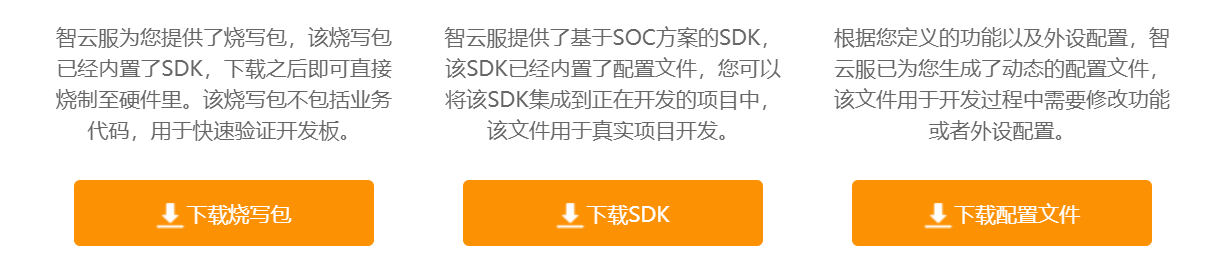


图 5.4 烧写包下载



图 5.5 硬件开发流程图

### 通信问题的解决

智云服平台提供了Android API文档以及Android SDK文件以及使用说明，如图5.6所示。在该文档中，可以看到开发板的数据会存储在一个名为DeviceDetailsEntity的实体类中，通过调用方法getRtData()获取实体类，由于其存储在ArrayList中，通过解析该数据，调用get()方法即可取得宠物项圈上的位置信息。Android也可通过API中的相应方法向开发板发送控制指令。



图 5.6 Android API文档

## 代码解释及接口信息

系统接口汇总：

**1.短信验证接口**

**2.注册接口**

3.**登陆接口**

4.配网接口

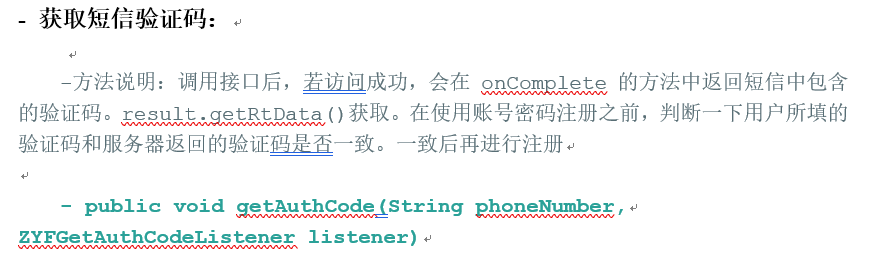
5.绑定、解绑设备接口

6.导航系统相关接口

### 接口具体代码

由于导航接口整合较为复杂，在5.4.2节给出具体说明。对于上文1-5中提出，在这里详细概述。

1.短信验证接口



2.注册接口

**使用账号密码进行注册：**

- 方法说明 ：注册后在 complete(**BaseEntity** result)回调方法中，通过 result.isRtState()获取注册状态,result.getRtMsg() 获取服务器返回的注册提示信息。

**- public void register(String phoneNumber, String password, ZYFRegisterListener listener)**

**ZYFSdk**.*getInstance*().register(**phoneNumber**, **psw**, **new ZYFRegisterListener**() {/\*\*/}

3.登陆接口

**-** 接口说明 ： 调用登陆接口的时候，在complete(**LoginEntity** result)回调方法中，通过 result.isRtState()获取登陆状态。根据状态判断是否登陆成功。

**- public void login(String phoneNumber, String password, ZYFLoginListener listener)**

**ZYFSdk**.*getInstance*().login(**phoneNumber**, **psw**, **new ZYFLoginListener**() {//}

调用登陆接口后返回的应用数据格式： json数据如下：

{"customerUser":

{"registerTime":"1541809577000","sid":72,"userName":"13338612187","userPassword":"E10ADC3949BA59ABBE56E057F20F883E","userPhone":"13338612187"},

"token":"c971f4c161434a8e95a98e5511959521"}

4.配网接口

AirLink 模式配网

**- public void airLinkSetNet(Context context, String ssid,**

**String pwd, AirLinkManager.AirLinkCallback airLinkCallback)**

**- ssid : 可用 WIFI 的名称**

**- pwd : 可用 WIFI 的密码**

**- airLinkCallback : airLink 配网的回调**

**ZYFSdk**.*getInstance*().airLinkSetNet(**context**,**ssid**,**pwd,new AirLinkManager.AirLinkCallback**() {/\*\*/}

SoftAP 模式配网

**- 第一步 ：将设备设置为 softAP 模式，然后调用下面方法，连接设备**

**- private void connectDeviceAp( Context context, String ssid, String pwd,WifiConnector.ConnectCallback connectCallback)**

**- ssid : 设备 WIFI 的名称**

**- pwd : 设备 WIFI 的密码**

**- connectCallback: 连接设备的回调**

**ZYFSdk**.*getInstance*().connectDeviceAp(**context**,**ssid**,**pwd,new WifiConnector.ConnectCallback**() {

**- 第二步 ：连接设备后，向设备发送可用 WIFI 的名称（SSID）和密码**

**- public void softApSetNet(String wifiSsid, String wifiPwd,**

**SoftApManager.SoftApCallback softApCallback)**

**- wifiSsid : 可用 WIFI 的名称**

**- wifiPwd : 可用 WIFI 的密码**

**- softApCallback: 发送 WIFI 信息的回调**

**ZYFSdk**.*getInstance*().softApSetNet(**context**,**ssid**,**pwd,new SoftApManager.SoftApCallback() {/\*\*/}**

5.绑定解绑设备接口

- 绑定设备

**- public void bindDevice(Context context,String deviceId, String customerId, ZYFBindDeviceListener listener)**

**- deviceId 设备的SN号**

**- customerId 登陆成功后返回的 sid**

说明：用户登录成功后，通过输入设备SN绑定设备，或者扫描二维码。

**ZYFSdk**.*getInstance*().bindDevice(**context**，**deviceId**,**customerId**, **new ZYFBindDeviceListener**(){

*@Override*

**public void** onComplete(**BaseEntity** result) {

}

*@Override*

**public void** onError(**Exception** e) {

}

});

- 回调方法onComplete参数说明

result.isRtState()

返回值类型 boolean : 绑定设备是否成功

result.getRtMsg()

返回值类型 String : 绑定结果提示信息

- 绑定设备

**- public void unBindDevice(Context context,String deviceId, String customerId, ZYFUnBindDeviceListener listener)**

**- deviceId 设备的SN号**

**ZYFSdk**.*getInstance*().unBindDevice(**context**，**deviceId**,

**customerId**, **new ZYFBindDeviceListener**() {

*@Override*

**public void** onComplete(**BindDevicesEntity** result) {

}

*@Override*

**public void** onError(**Exception** e) {

}

});

### 导航介绍

已在上文5中提出，在这里详细概述。

在使用导航功能时需要对Android进行相应配置，可以查看所使用导航公司提供的官方文档，这里不作介绍。下面介绍基础导航功能的调用，在程序中，有一个基础的导航类，名为BaseActivity，在该类中可以设置模拟行车速度（在测试时使用模拟导航方式需要该数据），语音功能的调用，路况查询，交通设施信息以及实时更新界面信息等功能，其他各种导航方式均继承自该类。在用户选择好导航方式（如用户选择了驾车导航）时，会跳转到驾车导航的类CustomCarActivity，如上所述，该类继承自BaseActivity，在该Activity初始化时，调用ZYF接口获取开发板上的经度和纬度，同时获取用户自身的经度和纬度在初始化成功调用的函数中，分别将获取到的经度和纬度赋给对应的实体类的数据项。即可进行该方式的导航。

需要注意的是由于安卓自身机制（Android4.0+）的改变，导航程序的运行都需要动态向用户申请位置权限，否则操作系统将不为导航提供移动端设备的经纬度。所以对用户的不给予位置权限行为需要做出相应处理。

我们目前已完成了骑行导航功能的移植。驾车导航中有躲避拥堵、不走高速、避免收费等策略选择，我们还未完成移植。

下面对骑行导航代码做简单介绍：

在导航初始化成功时，用requestPermission申请位置权限，并查看用户可用的位置提供器，通过位置提供器获取移动端设备的经纬度，再加载开发板的经纬度，并将他们分别封装未具体位置点，作为参数初始化骑行导航实例。其中需要注意的是，由于追踪宠物的导航不同于一般的寻找建筑物的导航，起点与终点都会发生变化，故在代码中需要监视地理位置信息的变化。通过

locationManager.requestLocationUpdates(locationProvider,updateTime,minDistance,locationListerner)进行。其中第二个参数为位置更新的时差，第三个参数的作用是在用户移动一定距离后进行路径的强制更新。以保证导航的准确性。其中，开发板位置信息的重新获取在locationListener的回调函数中进行。代码详见

com/example/linzhizhou/petchase12\_3/RideRouteCalculateActivity.java与同级目录下的BaseActivity.java文件。

### 服务端代码介绍

#### 基础配置

1.通过在服务端下的pom.xml配置需要下载的依赖包，服务端使用到的依赖包主要有（完整代码见配置文件）：

thymleaf模板引擎包（异步刷新时使用）

springboot配置相关的引擎包（start.spring.io）

hibernate相关配置包（hibernate.org）

数据库连接的驱动包

2.在中央仓库下载完这些包后进入application.properties对hibernate的连接进行信息进行配置（如下为部分配置信息，不包括数据库用户名密码等信息）。

#-----------------staticSource----------

spring.mvc.view.prefix=/static

spring.mvc.view.suffix=.html

spring.mvc.static-path-pattern=/static/\*\*

spring.resources.static-locations=classpath:/static,classpath:/public,classpath:/resources,classpath:/META-INF/resources

#-----------------dataSource------------

#spring.datasource.url=jdbc:mysql://localhost:3306/petchase?useSSL=false&serverTimezone=UTC

spring.datasource.url=jdbc:mysql://localhost:3306/petchase?useSSL=false&serverTimezone=UTC

spring.datasource.driverClassName = com.mysql.cj.jdbc.Driver

spring.datasource.max-active=20

spring.datasource.max-idle=8

spring.datasource.min-idle=8

spring.datasource.initial-size=10

server.port=80

#-----------------JPA-hibernate--------------

spring.jpa.database = MYSQL

spring.jpa.show-sql = true

spring.jpa.hibernate.ddl-auto = update

spring.jpa.hibernate.naming-strategy = org.hibernate.cfg.ImprovedNamingStrategy

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect

其中需要注意的是，由于要将其部署到云服务器上，为配合浏览器对80端口的省略，将服务端端口改为80，并在云服务端开放80端口。且由于数据库和服务器放在同一空间中，所以配置文件中连接数据库采用127.0.0.1的本机地址。

在云服务器运行jar包时将相关信息通过重定向命令输入到后台日志中，通过及时查看日志监视网站状态。

#### 用户请求处理

1. 登陆请求

用户在登陆时，安卓端通过HttpPost向服务端发送post请求，请求包含了用户的账号以及加密后的密码。服务端采用对象关系映射技术，利用hibernate将数据库的用户信息表映射到实体类中。通过对对象的操作来验证用户信息。其中，为了防止sql注入，在取用户数据的时候，只通过用户名取，不再验证密码，（避免hibernate生成"and 1=1"的sql注入代码），在取出用户记录后，再将该条数据的密码与用户传过来的密码进行比较。

1. 其他请求

用户在科普模块，以及修改部分设置信息时，由于信息相对不那么重要，且信息长度不长，所以安卓端采用get请求，服务端通过@RequestMapping以及@ResponseBody注解来响应该请求并返回相关信息。(注解含义见5.4.3.3)

1. 用户信息与设备信息关系处理

每个烧录成功的开发板都有一个唯一的SN号，要让用户能够绑定到自己的宠物，就需要对用户id与设备sn号进行绑定。在第三方接口中已经提供了绑定接口。在服务端我们再对该关系进行二次维护，来应对用户可能申诉无法绑定的情况。

#### 网页代码简介

服务端不仅仅为安卓端提供服务，还为用户提供了可视化的网页。

**对安卓端的响应**

对安卓端的响应利用了Spring MVC的控制器controller，控制器负责处理由DispatcherServlet 分发的请求，它把用户请求的数据经过业务处理层处理之后封装成一个Model ，然后再把该Model 返回给对应的View 进行展示。通过使用@Controller 标记一个类是Controller ，然后使用@RequestMapping 和@RequestParam 等注解获取请求与获取参数，用以定义URL 请求和Controller 方法之间的映射。当安卓端请求url时，加上@Responsebody注解，处理器方法的返回值通过HttpMessageConverters 转换之后写到HttpServletResponse 中。对安卓端的请求予以应答。如下为一个简单的注册应答，注册时由于已经在安卓客户端通过短信验证了用户的手机号码，客户端传过来的用户名必定唯一，故不再二次验证。当注册成功时，客户端收到"RegisterOK"后，即可进入app主界面。代码如下所示。

@RequestMapping(value = "/register")

@ResponseBody

public String register(@RequestParam(name = "uphone", required = true) String uphone, @RequestParam(name = "upass", required = true) String upass) {

user.setUid(Long.parseLong(uphone));

user.setUname(uphone);

user.setUphone(uphone);

user.setUpass(upass);

if (simpleService.insert(user)) {

return "RegisterOK";

} else {

return "RegisterFailed";

}

}

**对网页端UI资源的请求**

与响应安卓端的注解不同，请求网页资源不需要加上@ResponseBody注解（如上文所提到的，因为加上 @ResponseBody 后返回结果不会被解析为跳转路径，而是直接写入 HttpServletResponse中）。直接return即可请求网页资源（需要对拦截器进行配置，详见文档中的拦截器代码，代码解释见注释，在通过拦截器配置后，网页即可请求static下网页相关的css、js等代码以及templetes下的html等文件）。

@Configuration

@EnableWebMvc

public class MvcConfig extends WebMvcConfigurerAdapter {

@Override

public void addInterceptors(InterceptorRegistry registry) {

registry.addInterceptor(new ResourceInterceptor()).addPathPatterns("/\*\*").excludePathPatterns("/templates/\*","/static/\*");

super.addInterceptors(registry);

}

/\*\*

\* 修改springboot中默认的静态文件路径

\*/

@Override

public void addResourceHandlers(ResourceHandlerRegistry registry) {

//addResourceHandler请求路径

//addResourceLocations 在项目中的资源路径

//setCacheControl 设置静态资源缓存时间

registry.addResourceHandler("/static/\*\*").addResourceLocations("classpath:/static/"); registry.addResourceHandler(("/templates/\*\*")).addResourceLocations("classpath:/templates");

super.addResourceHandlers(registry);

}

在页面设计中使用Bootstrap动态响应式框架，并为移动端和PC端的不同界面特征做出了适配。

**错误拦截器**

通过自定义的拦截器，配置了自定义的错误界面。该拦截器对所有url进行处理，当出现错误时，将错误参数(url,以及model里的上下文参数message)传递给错误界面，界面根据参数显示更加友好的错误界面。

public class GlobalExceptionHandler implements ErrorViewResolver {

@Override

public ModelAndView resolveErrorView(HttpServletRequest request, HttpStatus status, Map<String, Object> model) {

ModelAndView modelAndView=new ModelAndView();

modelAndView.addObject("url",request.getRequestURL());

modelAndView.addAllObjects(model);**//message参数**

modelAndView.setViewName("error"); //查找templetes下的error.html文件

return null;

}

}

### 客户端与服务端交互部分代码

在安卓端实现客户端与服务端交互功能时，安卓端需要与服务器建立连接，由于程序的主线程需要对界面元素事件进行监听，此时必须开启多线程去建立和服务器的连接，连接的建立采用http的请求响应（request/response）模型。此时，若建立这次连接request的信息是核心且非常必要的信息时，主线程没有该信息从逻辑上来说就不该往下继续执行时，应让主线程等待子线程发出从服务器接收到的信息。以用户登陆为例子（用户输入用户名密码后，若没有接收到服务器信息，就不该跳转界面）。代码如下：

switch (view.getId())

{

case R.id.top\_back:

finish();

break;

case R.id.login\_miss:

startActivity(new Intent(this, MissActivity.class));

break;

case R.id.login\_btn:

new Thread(testLogin).start();

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

if(flag==1){

startActivity(new Intent(this, MainActivity.class));

finish();

}

else {

Toast t=Toast.makeText(getApplicationContext(),"用户名或密码错误",Toast.LENGTH\_SHORT);

t.show();

}

break；

default:

break;

}

灰底部分的代码为主线程代码，当用户单击了登陆按钮后，会开启一个子线程，并且主线程通过睡眠等待子线程（也可通过加锁），其中flag是一个标志变量，在子线程代码中会具体解释其含义。子线程代码如下：

Runnable testLogin=new Runnable() {

@Override

public void run() {

//加密密码

String encodePass;

encodePass=md5Decode16(passwd.getText().toString());

//通过域名访问服务端

strurl="http://lzzpros.cn:80/TestLogin?uphone=";

strurl=strurl+phone.getText().toString()+"&upass="+encodePass;

try {

URL url = null;

url = new URL(strurl);

//通过url建立连接

HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();

InputStreamReader in = new InputStreamReader(urlConn.getInputStream());

BufferedReader bufferedReader = new BufferedReader(in);

String result = "";

String readLine = null;

//遍历服务端返回的结果集合

while ((readLine = bufferedReader.readLine()) != null) {

result += readLine;

}

//如果登陆成果服务单会返回”OK”，见5.4.3节

if(result.equals("OK")){

flag=1;

}

//关闭连接

in.close();

urlConn.disconnect();}

catch(Exception e){

e.printStackTrace();

}

}

};

子线程具体任务见注释，通过注释可以大概了解整个连接建立的过程：**加密🡪准备连接信息，建立连接并发起request，服务器response，接收response信息并做相应处理**。（request和response被封装在了HttpURLConnection）中。接收到服务器返回的信息后，子线程不可以对界面UI进行处理，因为这是主线程的任务，界面的绘制为了保持结果的唯一性，必须单线程。此时，子线程将flag的值由0变为1，主线程在等待完毕后，检测flag的值，若为1将用户引导入主界面，若为0则在界面上通过Toast提示用户不能进行登陆。

注册逻辑大体与登陆类似，不过用户注册多了通过短信验证用户手机号码等过程，详见客户端文件夹下

.\src\main\java\com\example\linzhizhou\petchase\_12\_3\pet\ui\activity\ RegisterActivity.java文件，这里不再举例。另外，为了方便实时与用户交互信息，在科普模块下设置了信息实时更新模块，通过更新服务端的信息，即可在客户端动态刷新信息。代码详见客户端文件夹下

.\src\main\java\com\example\linzhizhou\petchase\_12\_3\FindFragment.java文件。

开发团队在开发过程中为了更好地去理解客户端与服务端交互过程，并没有采用okhttp、retrofit等框架。若是通过这些框架进行建立连接，框架会封装好线程的创建，无需再写开启子线程的代码。

# 完整功能介绍

## 网站功能

目前正在对开源商城UI界面进行更改优化，并对数据库进行相应更改。优化后商城可在网站上访问，也可以从app内的商城入口通过webview控件访问。无需二次开发手机商城。（目前尚未完善好该功能）在完善好该功能后，将考虑以出售宠物赠送追踪项圈的形式，吸引用户购买项圈。

### 已完善功能展示

网站网址为：lzzpros.cn,其中网站主界面如图6.1所示。



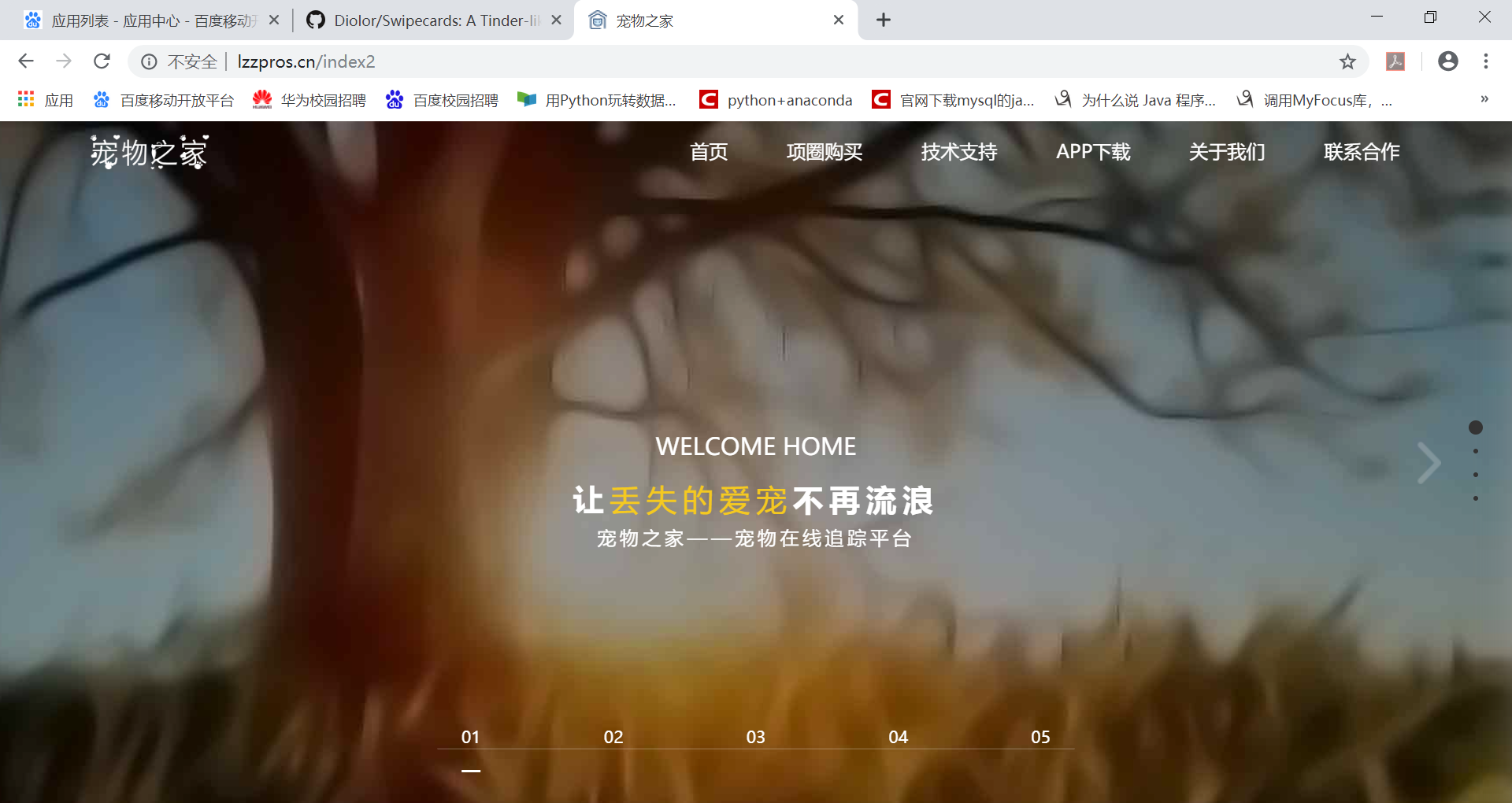


图6.1 网站首页图

网站为用户提供了如下功能：

在网站中用户可以看到软件的使用视频教程。如图6.2所示。

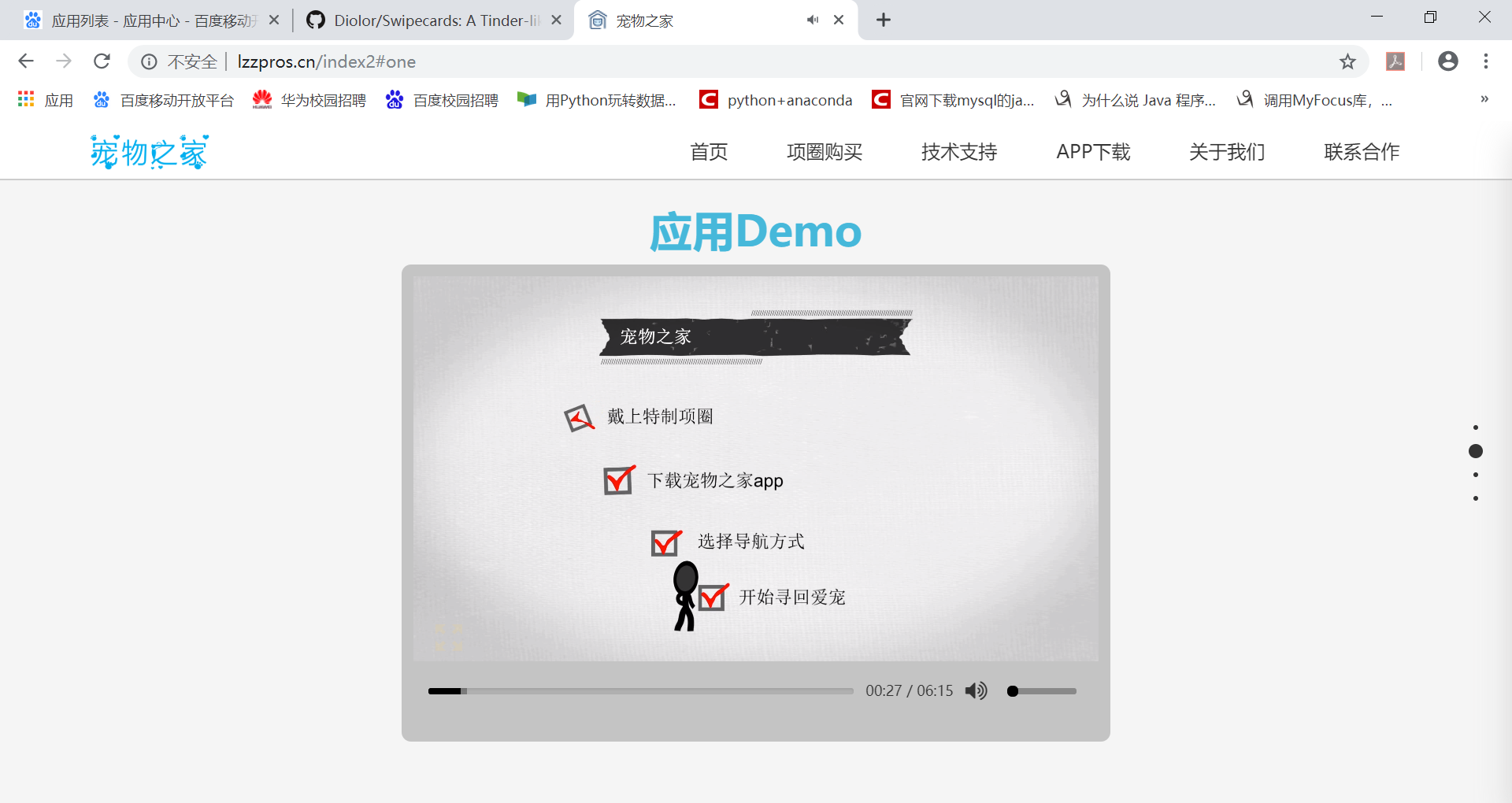


图6.2 应用Demo网站截图

购买宠物项圈（目前尚未完成该功能，该功能为一个商城的入口，通过该入口用户可以只选择购买项圈，也可以在商城中购买宠物赠送追踪项圈）预留界面入口如图6.7所示。

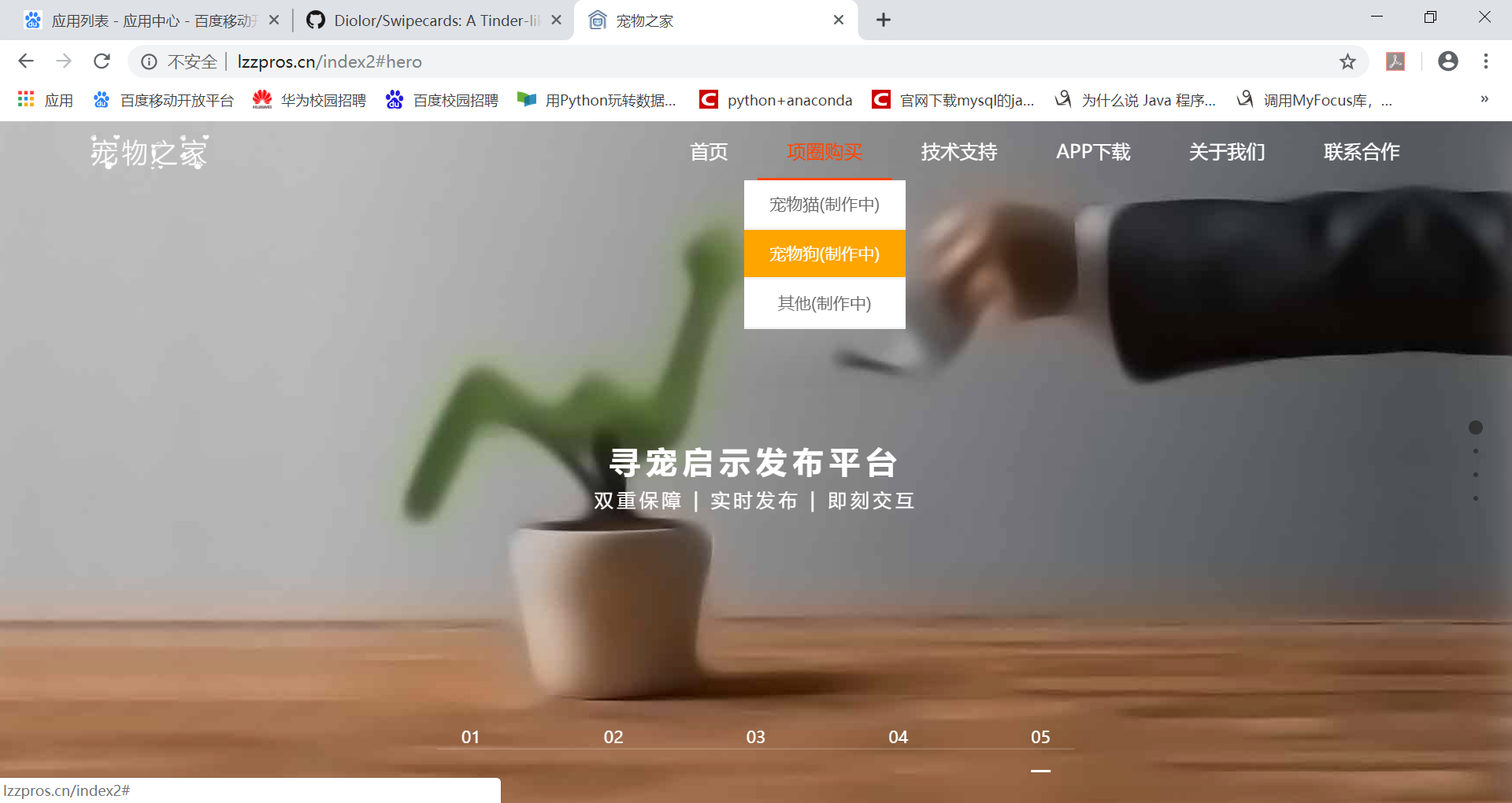


图6.7 项圈购买预留入口图

下载移动端应用（用户可以选择直接扫码下载或进入应用市场下载）。

查看应用使用指南等。如6.8所示。

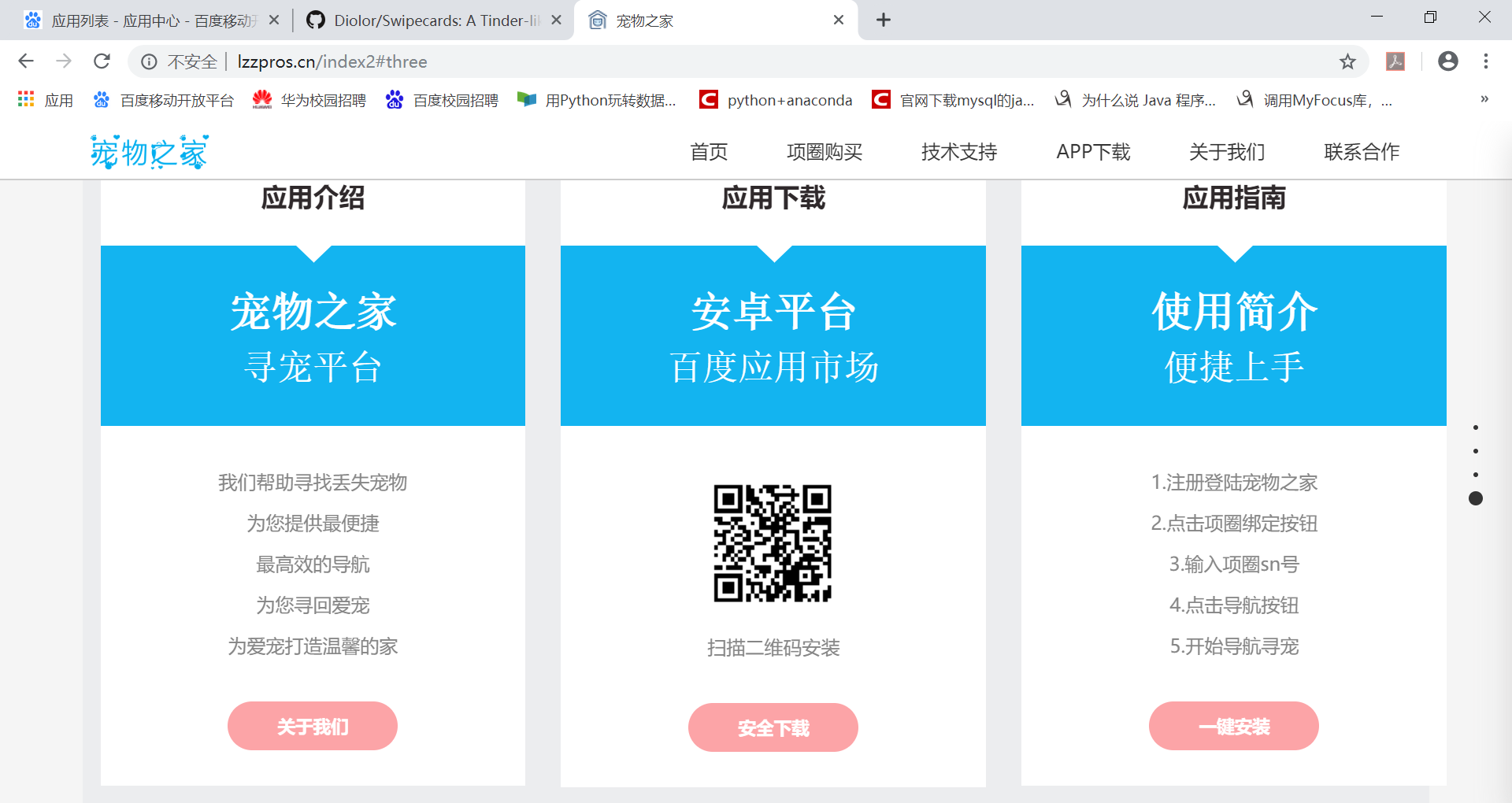


图6.8 应用指南

应用市场下载界面如图6.9所示。（截图较老，并非最新版本，最新版本请通过访问lzzpros.cn网站单击APP下载菜单栏下载）。

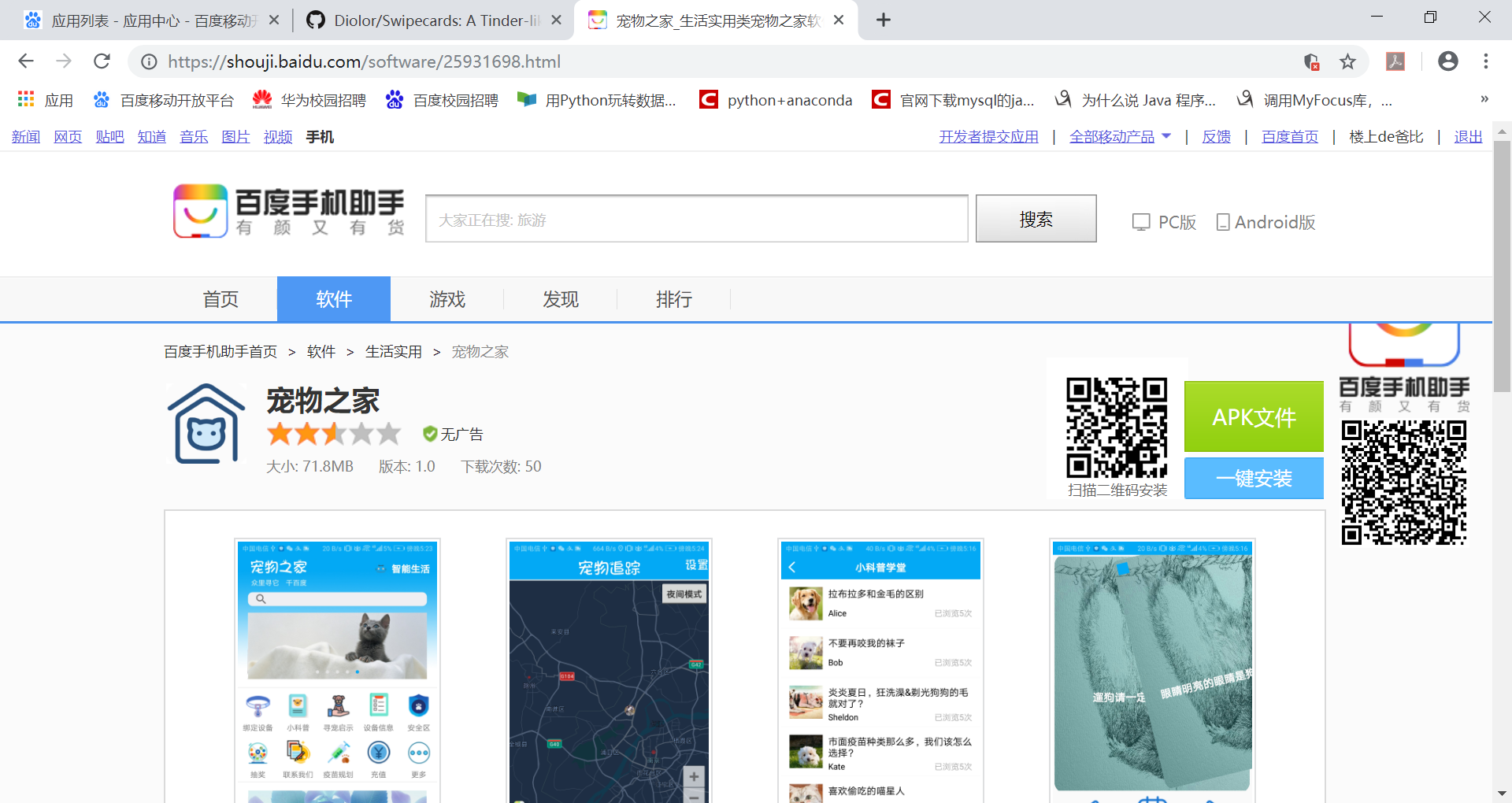


图6.9 应用市场截图

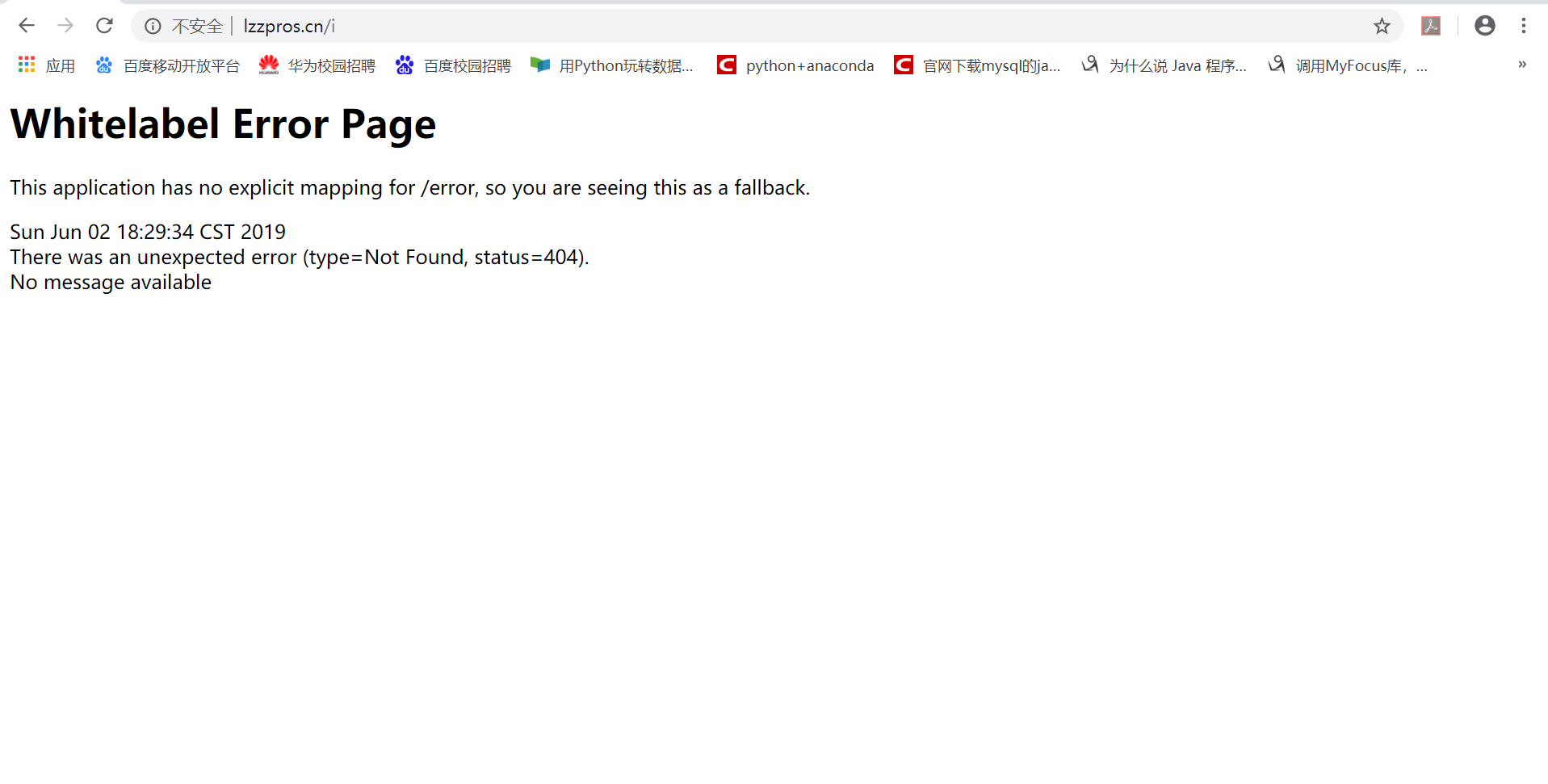
开发者信息界面，如图6.10所示。



图6.10 开发者信息界面图

### 错误拦截界面

为了不让网站显示如下不友好界面，通过拦截器对错误进行拦截。拦截后根据错误信息显示错误界面，具体效果对比如下。





## 移动端功能

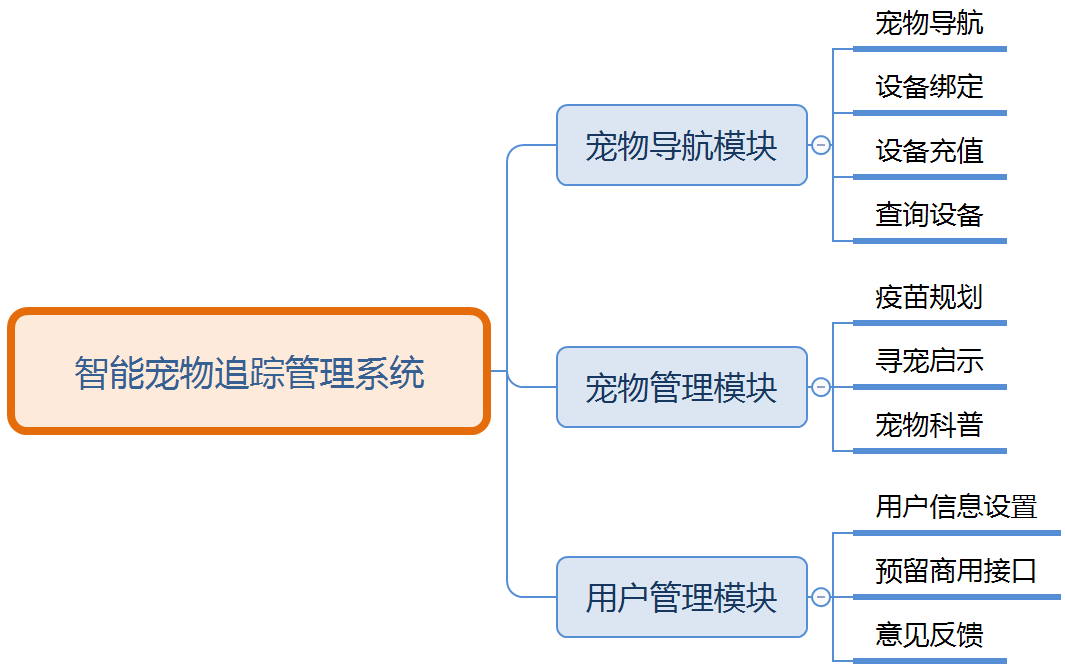


图6.11 模块调用关系图

系统分为三大主要功能模块：**宠物导航模块、宠物信息管理模块、用户信息管理模块。**如图6.11所示。其中导航模块是移动端的主要特色。除了导航外，还可以管理宠物打疫苗、发布和浏览丢失宠物、宠物知识科普等。

### 宠物导航模块

1. **宠物导航功能**

该功能中可以设置模拟行车速度、语音功能的调用、路况查询、交通设施信息以及实时更新界面信息等功能。在用户选择好导航方式（如用户选择了驾车导航）时，调用ZYF接口获取开发板上的经度和纬度，即获取用户自身的经度和纬度返回给App即可进行该方式的导航。如图6.12为提供的三种导航方式。

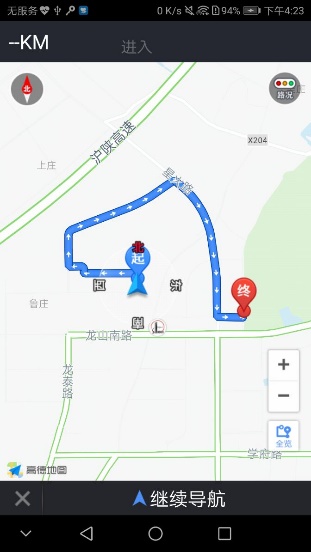
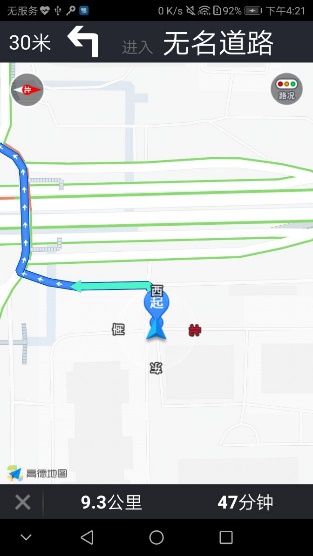
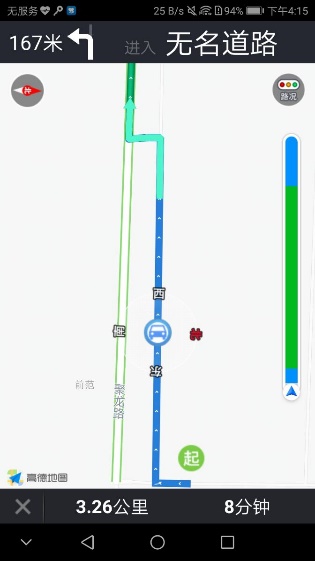
  

图6.12步行、骑行、驾车导航示例图

1. **绑定设备**

每个烧录成功的开发板都有一个唯一的SN号，安卓端可以通过SN号或扫描二维码来绑定设备。由于目前还在测试阶段，开发板数量不多，所以为设备绑定预留了接口，在进行量产时，只需对UI稍加改进，为数据库增加一张对应好SN号与用户id的表格即可，其他部分无需改动即可完善该功能，该功能初步界面如图6.13所示。



图6.13 设备绑定图

1. **设备充值**

要让宠物项圈能够发送位置信息，宠物项圈上必须要有物联网卡提供支持，物联网卡按流量计费，我们通过调用支付宝与物联网卡接口，为用户提供了充值功能，如图6.14所示。（由于百度不允许个人开发者预留支付接口，现已将该入口关闭）

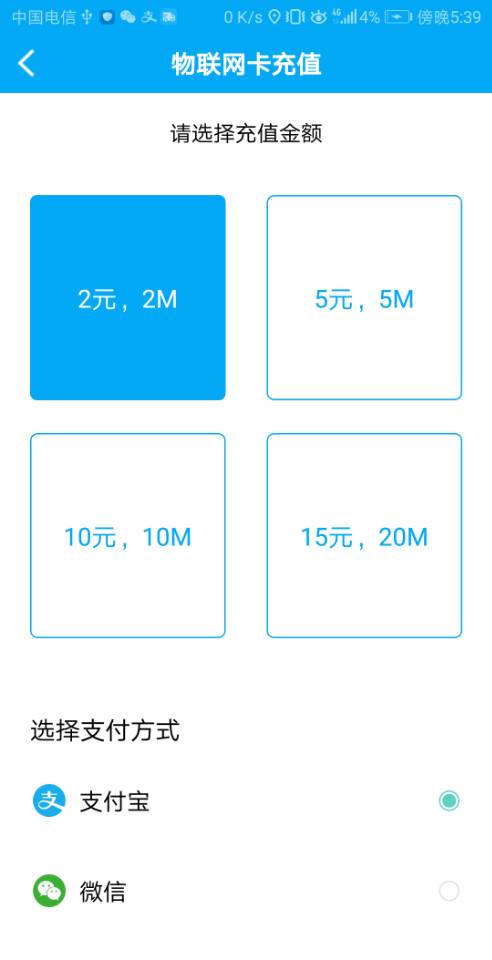


图6.14 物联网卡充值图

1. **设备查询**

用户可以在主页找到设备信息按钮，单击后可以实时显示宠物的经纬度位置信息，也可在导航界面可以实时看到宠物的图形化位置，在界面中哈士奇头像所标注的位置即宠物位置。界面如图6.15所示。

图6.15 设备信息查询图

### 宠物管理模块

1. **疫苗规划功能**

在该模块中，用户可以为宠物设定需要注射疫苗的时间，界面显示倒数日，当临近该日期时，系统会弹出对话框，提示用户及时带领宠物注射疫苗。界面如图6.16所示。

图6.16 疫苗规划UI图

**（2）宠物科普模块**

用户可以通过主页或者发现标签下进入两种效果的科普图，科普模块集成并改善了由Dilor提供的开源项目Swipecards。界面如图6.17所示。Dilor开源项目的原效果如图6.18所示。该开源项目地址：https://github.com/Diolor/Swipecards

图6.17 科普模块界面图

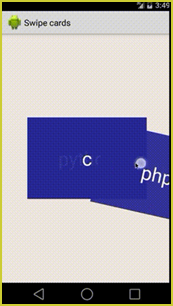


图6.18 Dilor’s Swipecards Image

### 用户管理模块

1. **用户信息设置**

用户只需填写手机号并绑定设备即可进行宠物追踪，用户可以选择在个人信息界面填写、修改、查看用户的个人信息，界面如图6.19所示。

图6.19 用户个人信息图

1. **预留商用接口**

App设计了后期商业运营预留了接口，目前以用户奖励功能为木块为例，奖励方式及奖励界面由后期广告商一同决定，奖励物品邮寄信息模块如图6.20所示。



图6.20中奖信息填写图

1. **意见反馈**

用户还可以在主页找到联系我们按钮，单击后可以像开发人员实时反馈信息，方便开发人员做出更好的改进。界面如图6.21所示。



图6.21 投诉建议图

# 系统测试与部署

## 测试

为了使应用具有更好的稳定性，我们开发团队使用了monkey runner编写脚本对软件进行了压力测试，但是由于安卓市场设备过多，一一测试几乎不可能实现。我们将应用提交到百度应用市场，产品通过了百度移动云测试中心的测试，已正式上线。如图7.1所示。



图7.1 中奖信息填写图

## 权限配置

在上文中已经提到需要特殊申请动态权限的情况。这里对其进行补充，如下为安卓所需要的所有权限。

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"/>

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_NETWORK\_STATE"/>

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.WAKE\_LOCK" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.RECEIVE\_SMS" >

</uses-permission>

<uses-permission android:name="android.permission.READ\_SMS" ></uses-permission>

其中，对于用户不同意就可能产生异常的权限已经进行动态申请，便于对异常进行处理。对部分权限进行进一步的说明如表7.1所示：

|  |  |
| --- | --- |
| 权限 | 用途 |
| INTERNET | 允许程序打开网络 |
| ACCESS\_WIFI\_STATE | 允许程序访问WiFI网络状态信息 |
| ACCESS\_NETWORK\_STATE | 允许程序访问有关GSM网络信息 |
| CHANGE\_WIFI\_STATE | 允许程序改变WiFi连接状态 |
| CHANGE\_WIFI\_MULTICAST\_STATE | 允许程序改变WiFi多播状态 |

表7.1部分权限具体说明表

## 完成情况

★可行性分析 完成

★需求分析 完成

★宠物商城数据库设计 进行中

★导航数据库数据获取 完成

★UI设计基本完成 完成

★安卓端开发 完成

★社交服务端开发 完成

★宠物商城界面设计 部分完成，入口尚未开放

## 困难总结

### 接口版本问题

智云服平台提供的Android API文档和Android SDK文档的版本更新不及时，过于老旧，导致调用时接口数量以及参数位置无法对应，多次调参均无法改正，在经过联系智云服技术支持后得以解决。

### 虚拟机上的导航功能问题

由于一开始缺乏经验，在虚拟机上进行导航功能的调试，导致虚拟机黑屏，无法调出地图，在通过查询资料寻找黑屏原因后，发现在虚拟机上不能运行导航（由于电脑没有GPS模块与SIM卡以及其他一些列原因），最后换成真机调试后问题得到解决。

### 调用外部库问题

安卓开发调用了由Diolor在Github上开源的Swipecards动态效果，由于Android版本不兼容，导致闪退，在充分查询网上资料，多次调整版本改变而带来的代码后，问题得以解决。

网页端lzzpros.cn/index在IE中由于调用了某个兼容性不高的库，导致在该界面在IE浏览器下可能会有部分黄圈，问题正在修复中。

## 经验总结



在开发过程中遇见了许多的问题，在解决这些问题的过程中，我们开发团队总结出了一些经验，作为日后团队的自我反省。

★在需求不明确时，团队中一定要有人从不同的视觉看待这个开发过程，只从技术角度开发很可能会偏离一开始的目标。从不同视觉看待开发，往往会有创新点的提出。

★团队协作>个人

★团队中每个开发者都必须掌握好代码管理托管的平台是开发的前提。

★在小团队中，每个开发阶段产出一些文档来标志着这个开发阶段的结束，有利于鼓舞士气，加快开发进度。

# 参考文献

[1].中商产业研究院. 2018年宠物行业市场前景研究报告[R].

<http://www.askci.com/news/chanye/20180613/1146121124635_2.shtml>.

[2].[潘爵雨.胡斌杰]基于RFID的室内定位技术及其应用研究[R]. 华南理工大学

[3].[宋欣.钱良]多传感融合的室内

定位技术研究[R].上海交通大学

[4]. [汪飞]gpsOne混合定位技术的研究与应用-全球卫星定位技术.

南京邮电大学

[5] [张肖].基于GPS\_GPRS的定位系统在物探测量中的应用研究.[R].

华北电力大学

[6][汪飞] gpsOne混合定位技术AGPS定位[R].南京邮电大学

[7]. 张超. 蓝牙个域网改型及芯片化实现技术研究[R.] 西安电子科技大学

[8].姜梦兰.基于消息中间服务可靠性保证方案的研究与实现[硕士学位论文].成都.电子科技大学.2010

[9].Lee S,Kim H,Hong D,Ju H.Correlation analysis of MQTT loss and delay according to QoS level.Information Networking(ICOIN).Bangkok.

2013.714-717

[10].袁远.基于Android平台端到端即时通信系统的分析与设计[硕士学位论文].北京.北京邮电大学.2012

[11].王楠，宋飞，周华春.一种基于Android平台的即时通信方案.计算机应用与软件,2013年，（4）：107-109,148.

[12].关庆余,李鸿彬,于波.MQTT 协议在Android 平台上的研究与应用[J].Research and Development 研究开发,2014年，（第 23 卷第 4 期）

[13].智云服-微智.203CS开发平台使用说明书.

<http://www.hizyf.com/components/coreProducts/info.html?sid=35>.2018年3月

[14].智云服. WZ203CS-V3.0开发板support文档(20180521).