



本科生毕业论文（设计）

Undergraduate Graduation Thesis (Design)

题目 基于 Android 的宠物日记 App 设计与实现
Title

院系 数据科学与计算机学院
School (Department)

专业 软件工程
Major

学生姓名 陈亚楠
Student Name

学号 16340041
Student No.

指导教师（职称） 刘聪（副教授）
Supervisor (Title)

时间：2020 年 03 月 25 日

Date: March 25, 2020

表一：毕业论文（设计）开题报告
Form 1: Research Proposal of Graduation Thesis (Design)

论文（设计）题目

Thesis (Design) Title: 基于 Android 的宠物日记 App 设计与开发

(简述选题的目的、思路、方法、相关支持条件及进度安排等)

(Please briefly state the research objective, research methodology, research procedure and research schedule in this part.)

一、目的

随着社会经济的发展，人们的物质生活得到了很大的改善。同时，伴随着人口老龄化、单身群体数量增加等社会问题，宠物已经被越来越多的人接受，成为人们情感寄托的一种载体，同时宠物 APP 也应运而生。

通过查阅相关文献，我们发现宠物 APP 的产品定位主要为两方面：社交与电商。这两方面主要针对养宠人士，确实方便了养宠人士的日常需求。但宠物 APP 的发源与发展更大程度上应当是以宠物作为中心基本点。围绕关注与关心宠物进行设计与开发。

关于宠物 APP 的设计，我们的关注点应当为人与宠物的关系，以及针对宠物健康、宠物医疗等切乎宠物本身的实际需求而进行分析设计。比如说，宠物健康信息的记录与存储。目前，宠物医院提供的宠物信息记录服务主要以纸质的疫苗本为记录载体，这样的记录形式存在易丢失、易损坏的问题；除此之外，包括宠物疾病史、药品使用、驱虫等健康信息均不在宠物医院提供的记录信息之列。

二、思路与方法

本次毕业设计希望通过以下功能解决上述提到的宠物 APP 在设计方面遇到的问题：

(一) 创建宠物日记：

通过由养宠人士创建日记来记录自己与宠物的生活日常，将宠物 APP 的产品关系由社交指示的人与人的关系转移到人与宠物的关系，使宠物 APP 的关注点为“宠物”这一核心要素。

(二) 建立宠物健康档案：

记录宠物的基本健康信息，包括疾病记录、使用药品信息、疫苗接种记录、

驱虫记录等，解决纸质记录导致信息容易损坏与丢失的问题，供宠物医生与宠物家长参阅。

三、相关支持

(1) 宠物 APP 相关研究论文。

(2) Android 平台程序开发技术。

四、进度安排

2019 年 11 月：确定毕业设计选题。

2019 年 12 月：进行程序系统分析与设计，完成相关文档。

2020 年 1 月-2020 年 2 月；进入程序编写阶段，完成整个程序的代码部分。

2020 年 3 月 - 2020 年 4 月；完善程序与文档，完成论文初稿修改并定稿。

Student Signature:

Date:

指导教师意见

Comments from Supervisor:

通过设计一个关注宠物健康的 APP，锻炼学生的软件设计和开发能力。该软件具有一定的实用价值，和一定的新颖性（在开题的时候找不到类似软件的存在）。软件开发难度适中。

1.同意开题

2.修改后开题

3.重新开题

1.Approved()

2. Approved after Revision ()

3. Disapproved()

Supervisor Signature:



Date:

表二：毕业论文（设计）过程检查情况记录表
Form 2: Process Check-up Form

指导教师分阶段检查论文的进展情况（要求过程检查记录不少于 3 次）

The supervisor should check up the working process for the thesis (design) and fill up the following check-up log. At least three times of the check-up should be done and kept on the log.

第 1 次检查 (First Check-up) :

学生总结

Student Self-summary:

首先确定了以宠物为主的选题方向；通过查阅相关文献，确定了具体的选题内容与 APP 功能定位。

指导教师意见

Comments of Supervisor:

功能单一，建议再次考虑功能定位，提出比较完善的、可行性较高的功能。

第 2 次检查 (Second Check-up) :

学生总结

Student Self-summary:

进行 APP 的系统分析与设计，完成了相关 UML 建模。

指导教师意见

Comments of Supervisor:

已经完成软件的设计，建议功能上可以增加病历等记录功能。

第 3 次检查 (Third Check-up) :

学生总结

Student Self-summary:

完成了程序代码编写，使用了 Android Jetpack 工具库以及其相关的组件，数据库方面使用 Room 持久性库而非传统的 SQLite 数据库。

指导教师意见

Comments of Supervisor:

程序可以正常运行，设计功能均已实现，请开始论文的编写。

第 4 次检查

Fourth Check-up

学生总结

Student Self-summary:

针对程序与文档进行了修改与补充，修改了论文初稿，校正了格式问题，最终定稿。

指导教师意见

Comments of Supervisor:

论文格式符合要求，可以定稿。

学生签名 (Student Signature) :

日期 (Date) :

指导教师签名 (Supervisor Signature) :



日期 (Date) :

总体完成情况
(Overall
Assessment)

指导教师意见 Comments of Supervisor:

- 1、按计划完成，完成情况优 (Excellent) : ()
- 2、按计划完成，完成情况良 (Good) : (√)
- 3、基本按计划完成，完成情况合格 (Fair) : ()
- 4、完成情况不合格 (Poor) : ()

指导教师签名 (Supervisor Signature) :



日期 (Date) :

表三：毕业论文（设计）答辩情况登记表
Form 3: Thesis Defense Performance Form

答辩人 Student Name		专 业 Major	
论文（设计）题目 Thesis (Design) Title			
答辩小组成员 Committee Members			
<div>答辩记录 Records of Defense Performance:</div> <div></div> <div>记录人签名（Clerk Signature）：</div> <div>日期（Date）：</div>			

学术诚信声明

本人所呈交的毕业论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料均真实可靠。除文中已经注明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的作品或成果。对本论文的研究作出重要贡献的个人和集体，均已在文中以明确的方式标明。本毕业论文的知识产权归属于培养单位。本人完全意识到本声明的法律结果由本人承担。

本人签名：陈亚楠

日期：2020 年 03 月 25 日

Statement of Academic Integrity

I hereby acknowledge that the thesis submitted is a product of my own independent research under the supervision of my supervisor, and that all the data, statistics, pictures and materials are reliable and trustworthy, and that all the previous research and sources are appropriately marked in the thesis, and that the intellectual property of the thesis belongs to the school. I am fully aware of the legal effect of this statement.

Student Signature: Yanan Chen

Date: March 25, 2020

摘 要

根据《2019 年中国宠物行业白皮书》（消费报告）显示，在 2019 年，我国全国城镇宠物犬猫数量已经达到 9915 万只，相比 2018 年增长 8.6%；2019 年全国城镇宠物犬猫消费市场规模达到 2014 亿元，同比增长 18.5%；我国宠物行业发展态势良好，同时，宠物智能手机应用程序(Application，以下简称 APP)数量逐渐增多，功能趋加多样。然而，综合比较目前市场上的宠物 APP 可以发现，大多数 APP 在产品设计上舍本逐末，忽略宠物本身，而更多去关注人与人之间的社交。

本文基于这一问题，采用 Android Jetpack 工具库的推荐应用架构，使用该工具库的 Navigation、Fragment 等相关组件工具，以及通过使用 Room 持久性库实现本地数据存储，分析、设计与实现了 Android 平台手机应用程序“喵呜”。该程序以日记的形式作为表达载体，建立人与宠物的独立空间，关注人与宠物之间的关系；同时，辅以建立宠物档案等功能，关注宠物本身，回归宠物 APP 的设计初衷。

关键词： 宠物；Application；Android；Android Jetpack；Room

ABSTRACT

According to the 2019 white paper on China's pet industry (consumption report), the number of urban pet dogs and cats in China reached 99.15 million in 2019, up 8.6 percent year-on-year. In 2019, the consumption market of urban pet dogs and cats in China reached RMB 2,014 billion, with a year-on-year growth of 18.5%. China's pet industry has a good development trend. At the same time, the number of pet smartphone applications (hereinafter referred to as apps) is gradually increasing and their functions are becoming more and more diversified. However, a comprehensive comparison of the current pet apps on the market shows that most of them neglect the essence in product design, ignore the pets themselves and pay more attention to the social interaction between people.

Based on this problem, this paper adopts the recommended application architecture of the Android Jetpack tool library, USES the Navigation, Fragment and other component tools of the tool library, and USES the Room persistence library to achieve local data storage, analyzes, designs and develops the mobile application "meow" on the Android platform. This program takes the form of diary as the center, establishes the independent space between people and pets, and pays attention to the relationship between people and pets. At the same time, with the help of functions such as establishing pet files, we will pay attention to the pets themselves and return to the original intention of the pet APP.

Keywords: Pet; Application; Android; Android Jetpack; Room

目 录

第一章 概述	1
1.1 APP 设计背景	1
1.2 问题描述	1
1.3 本文的工作	2
1.4 论文结构简介	2
第二章 程序开发相关技术综述	4
2.1 Android Jetpack	4
2.2 LiveData	5
2.3 Navigation	5
2.4 Room 持久性库	6
第三章 系统需求分析	8
3.1 功能分析	8
3.2 系统用例图	8
第四章 系统设计	10
4.1 应用架构设计	10
4.1.1 MVVM	10
4.1.2 系统架构设计	11
4.2 数据库设计	13
4.2.1 数据库概念设计	13
4.2.2 数据表设计	14
第五章 系统实现与测试	16
5.1 系统实现	16
5.1.1 开发环境	16
5.1.2 功能模块实现	16
5.2 系统测试	21
5.2.1 功能测试	21
5.2.2 测试结果分析	22
第六章 总结与展望	23
6.1 总结	23
6.2 展望	23
参考文献	25
致 谢	26
附 录	27

附录 A.....	27
附录 B.....	29

第一章 概述

1.1 APP 设计背景

随着人口老龄化、丁克家庭数量增多、单身人口数量增加等社会问题出现，宠物由于具有能够缓解人类压力、改善人类健康、寄托人类情感的特点，愈发被人类所接受与喜爱。同时，伴随着经济的快速发展，宠物的数量与生存质量得到普遍提升。《2019 年中国宠物行业白皮书》数据显示，在 2019 年，我国城镇宠物猫与宠物犬的数量总计为 9915 万只，同比增长 8.4%；全国城镇宠物犬猫消费市场规模达到 2024 亿元，城镇养宠总体消费市场规模同比增长 18.5%；2019 年单只宠物年消费金额达到 5561 元。目前，我国宠物行业正处在一个快速发展的时期，发展前景广阔。

在当今“互联网+”的时代背景下，传统宠物行业的服务模式发展稍显滞后，难以满足人们的需求，互联网与宠物行业服务的融合发展大势所趋。其中，由于移动互联网具有便利性、高效性的特点，以手机 APP 为主的移动互联网宠物服务，正在为养宠人士带来非常大的方便。目前，全球智能手机应用系统主要由 7 类构成：Android 系统、苹果系统、微软系统、黑莓系统、塞班系统、三星系统等；其中，Android 是一种基于 Linux 内核的自由与开放源代码的操作系统，由 Google 与开放手持设备联盟领导及研发。根据相关数据显示，Android 系统使用数量占据全球第一，在全球智能手机市场份额中占比为 76%，占据中国市场 90% 的份额^[1]。

1.2 问题描述

目前，我国宠物市场繁荣发展，宠物类 APP 市场发展前景广阔，宠物类 APP 数量增多，功能日趋完善，能够满足养宠人士的基本需求。但从整体现状来看，宠物类 APP 市场发展尚未达到饱和，目前市场上暂未出现行业领头者，宠物类 APP 的设计与开发仍存在问题。

目前的宠物类 APP 同质化严重，产品定位大同小异，多侧重于“社交”和“电商”，产品在关系表达上忽视了“宠物”这一核心要素，忽视了人与宠物之间的关

系，仍然更多地关注人与人之间的关系。

宠物医疗是宠物产业中的核心节点，在美国，宠物医疗在宠物行业市场容量中的占比为 50%，而在我国，宠物医疗仅占比整个宠物产业的 25%，尚处于起步阶段，因此我国宠物医疗市场规模在整个宠物行业中具有较大的发展空间^[2]。尽管如此，目前宠物行业市场上仍缺少具备完善功能的关注宠物医疗的宠物 APP，缺少能够系统记录宠物身体状况的宠物 APP。

1.3 本文的工作

根据目前宠物类 APP 存在的问题，本文分析、设计、实现了一个 Android 平台手机 APP “喵呜”，通过以下两个功能来解决上述提到的关于宠物类 APP 的两个问题：

(1) 创建宠物日记：用户通过在这里发布文字或照片，记录与宠物相处的生活点滴。

(2) 建立宠物档案：用户通过该功能记录宠物的基本信息，包括宠物姓名、性别、出生日期、疾病诊断历史、用药历史等。

通过上述功能，“喵呜”APP 能够帮助养宠人士系统地关注与了解宠物的身体状况，建立人与宠物的关系空间，以使宠物能够更好地生活为目标，将宠物类 APP 开发的中心转移到“宠物”这一核心要素上。

1.4 论文结构简介

本文总共由五个章节组成，每章的主要内容介绍如下：

第一章为论文的概述部分。该章分为四小节，依次介绍“喵呜”APP 的设计背景、“喵呜”APP 要解决的目前宠物类 APP 存在的问题、本文完成的工作以及本论文的内容结构。

第二章介绍了“喵呜”APP 在开发过程中使用的相关程序开发技术，包括 Android Jetpack 工具库、LiveData 数据存储类、Navigation 导航组件、Room 持久性库等。

第三章为本论文的需求分析部分，该部分进行了 APP 系统功能分析并使用

用例图表示。

第四章对“喵喵”APP 的系统设计进行了阐述，介绍了 MVVM (Model-View-ViewModel) 软件架构模式，本应用程序的应用架构设计以及数据库的设计。

第五章为“喵喵”APP 的系统实现部分，在这一章介绍了开发环境、系统各功能模块的实现、以及系统测试等。

第六章为本次程序开发进行了总结与展望，主要内容为总结本文的工作、在本次程序开发中的收获以及对“喵喵”APP 的未来开发期望。

第二章 程序开发相关技术综述

本章将介绍“喵呜”APP 在程序开发过程中使用的工具与技术：Android Jetpack 工具库、LiveData 数据存储类、Navigation 导航组件、Room 持久性库等。

2.1 Android Jetpack

Android Jetpack 是 Google 在 2018 年发布的一套库、工具和指南，它包含的组件能够帮助开发者遵循最佳做法，避免编写模板代码，简化复杂任务，使得开发者更加集中地关注所需代码，加速开发，构建高质量的强大应用。

Android Jetpack 可以管理繁琐的 Activity（例如后台任务、导航、生命周期管理等），因此开发者能够更加关注提升应用本身；除此之外，Android Jetpack 包含与平台 API 解绑的 androidx 软件包库，这表示 Jetpack 可以提供向后兼容性，从而减少程序崩溃与内存泄漏问题；并且，由于 Jetpack 始终保持比 Android 平台更高的更新频率，开发者总是可以获取并使用最新、最好的 Jetpack 组件版本。

Android Jetpack 组件可以单独使用，也可以协同工作。其组件覆盖四个方面：基础组件、架构组件、行为组件和界面组件，图 2.1 展示了完整的 Android Jetpack 组件构成。

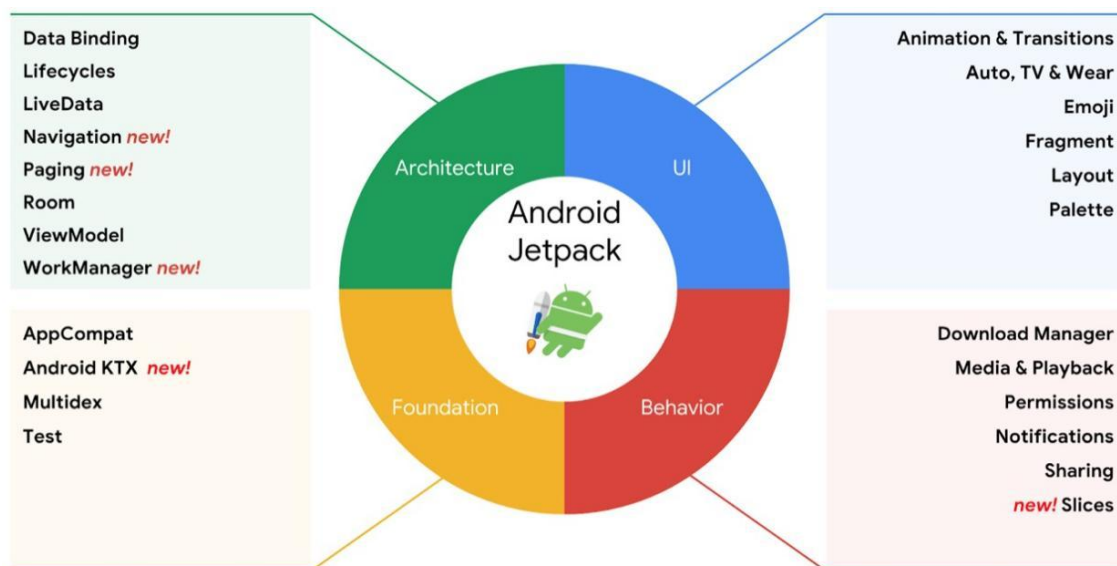


图 2.1 Android Jetpack 组件构成（图片来源：Android Developer.Android Jetpack）

2.2 LiveData

LiveData 是一种可观察的数据存储类，它能够遵循例如 Activity、Fragment、Service 等应用组件的生命周期，它所具有的这种特殊的生命周期感知能力，使得只有处于活跃生命周期状态的应用组件观察者能够接收到来自 LiveData 的更新通知。

在开发过程中使用 LiveData 具有以下七个优点：

(1) 确保界面始终与数据状态相关联：LiveData 遵循观察者模式，它能够在生命周期发生变化时通知观察者，因此开发者可以将代码整合到观察者对象中然后更新界面。观察者可以在每次发生改变时更新界面，而不是在应用数据每次发生改变时更新界面。

(2) 无内存泄漏发生：由于观察者与生命周期对象绑定，因此当与观察者相关联的生命周期被销毁时，它们能够及时进行自我清理。

(3) 不会因 Activity 停止而导致程序发生崩溃：如果观察者的生命周期处于不活跃状态，则不接受任何 LiveData 事件。

(4) 无需手动处理生命周期：界面组件仅观察相关联的数据，而不会去停止或恢复观察行为。LiveData 在观察时总是能够感知到相关的生命周期状态变化，因此它可以自动管理所有这些操作。

(5) 数据始终保持最新状态：如果某个组件的生命周期由活跃状态变为非活跃状态，LiveData 可以使得它在再次变为活跃状态时依然接收到最新的数据。

(6) 适当的配置更改：如果一个 Activity 或 Fragment 由于配置更改而被重新创建时，它依然可以通过 LiveData 获取最新的可用数据。

(7) 能够实现共享资源：开发者能够通过使用单例模式扩展 LiveData 对象来封装系统服务，然后在应用中共享它们。

2.3 Navigation

导航是指一种允许用户在应用程序中导航、进入以及退出不同内容片段的交互行为刚发。

Android Jetpack 中的导航组件由导航图、NavHost、NavController 三个关键

部分组成。导航图是在一个集中位置包含了所有与导航相关的信息的 XML 资源，这包括了应用中所有称为“目标”的独立内容区域以及用户可以通过应用获取的所有可能路径。NavHost 是一个从导航图中展示“目标”的空容器。NavController 是在 NavHost 中管理应用导航的对象，当用户想要在应用中移动时，NavController 可以安排在 NavHost 中的“目标”内容的交换。当在应用中导航时，用户告诉 NavController 他想沿导航图中的特定路径导航至特定目标，或者直接导航至特定目标，然后 NavController 就会在 NavHost 中显示相应目标。

Android Jetpack 的导航组件具有以下七个优点：

- (1) 可以处理 Fragment 事务。
- (2) 能够默认处理前进或后退操作。
- (3) 提供标准化资源以供动画和转换使用。
- (4) 实现与处理深层链接。
- (5) 包含例如抽屉式导航、底部导航等的导航界面模式，开发者只需再完成较少的工作。
- (6) 通过使用 Safe Args 来传递安全的数据，Safe Args 是 Navigation 组件的一个 Gradle 插件，该插件可以生成简单的 object 和 builder 类，以便通过类型安全的方式浏览和访问任何关联的参数。
- (7) 用户可以将 ViewModel 的范围限定为导航图，以共享导航图的“目标”之间的与界面相关的数据。

2.4 Room 持久性库

Room 是 Google 提供的一个 ORM (Object Relational Mapping) 数据库，它在 SQLite 上提供了一个抽象层，对原生的 SQLite API 进行了封装，使得开发者在充分利用 SQLite 的强大功能的同时，可以获得更强健的数据库访问机制。

Room 能够在设备上创建应用数据的缓存，并且该缓存充当应用的单一可信来源，因此用户可以在应用中查看关键信息的一致副本。用户可以在离线状态下浏览相关内容即使用户的设备无法访问网络，然后在设备接入到网络后，Room 可以将用户发起的内容更改同步到服务器。

与 SQLite 相比，Room 具备以下三个优点：

(1) Room 使得 SQL 查询在编译时就执行验证, 避免了任何运行时错误可能导致的程序崩溃。

(2) Room 具有更少的模版代码, 程序代码更加简单明了。

(3) Room 与 LiveData 集成, 当数据库更新时, Room 会生成更新 LiveData 对象所需的所有代码, 在必要时, 生成的代码会在后台线程上异步运行查询, 这种模式有助于保持界面中显示的数据与存储在数据库中的数据同步。

Room 有三个主要组件构成: 数据库、Entity、DAO (Data Access Objects) 。其中, 数据库包含了数据库所有者, 该组件是接入应用已保留的持久关系型数据库的底层连接的主要接入点。Entity 指的是数据库中的表。DAO 包含了用于访问数据库的方法。这三个组件之间的关系如图 2.2 所示。

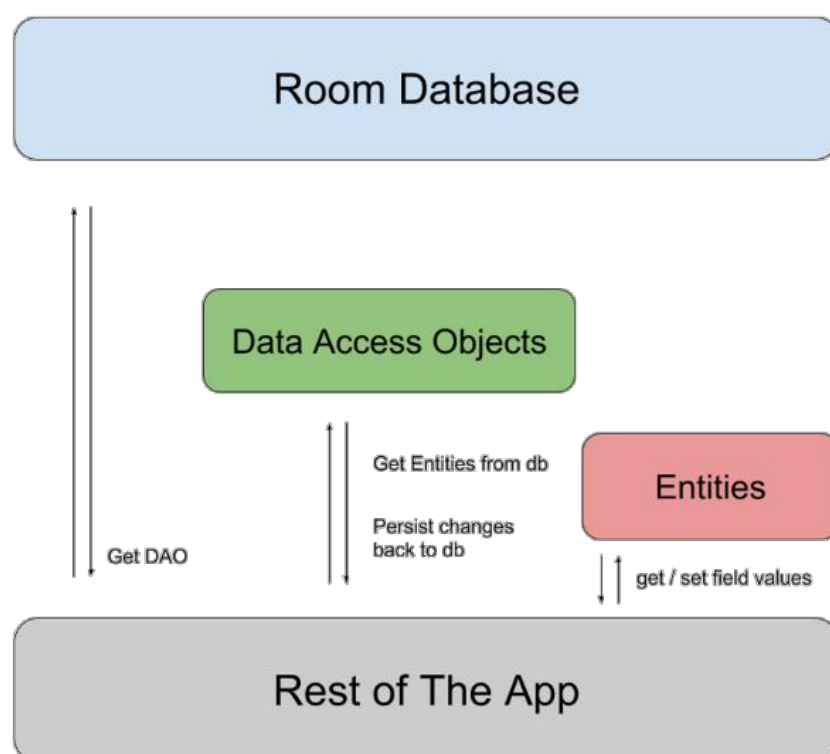


图 2.2 Room 架构图 (图片来源: Android Developer.Save data in a local database using Room)

应用通过使用 Room 数据库来获取与该数据库关联的 DAO; 然后, 应用通过使用每个 DAO 来获取 Entities, 并且当 Entities 发生更改时, 应用可以通过 DAO 将这些 Entities 的更改保存回 Room 数据库; 最后, 应用通过使用 Entities 的 getter 或 setter 方法来获取或设置数据库各个表的不同字段对应的值。

第三章 系统需求分析

3.1 功能分析

“喵呜”APP 设计目的主要是为了帮助养宠人士关注宠物健康，创建养宠人士与宠物的专属空间，拉近人与宠物的关系，因而它具备创建宠物日记、建立宠物健康档案两项主要内容。

(1) 创建宠物日记:

养宠人士以日记的形式记录与宠物的日常生活，创建了一个只包含养宠人士个人与宠物的单独空间，同时，通过时间线养宠人士也可以直观感受宠物的成长过程，增强其与宠物之间的亲密度。

该功能模块包含如下的具体功能：创建日记、修改日记、删除日记、分享日记到其他平台。

(2) 建立宠物健康档案:

在手机端建立宠物健康信息的电子档案，避免了纸质记录易磨损、易丢失、难携带的问题。同时，电子档案记录清晰明了，可以长期记录与查看。

该功能模块可以创建宠物档案、删除宠物档案、以及编辑宠物档案的以下信息：姓名、性别、出生日期、疾病诊断记录、药品使用记录、疫苗注射记录、针对猫犬宠物的驱虫记录等。

3.2 系统用例图

图 3.1 展示了本应用程序的系统用例图。

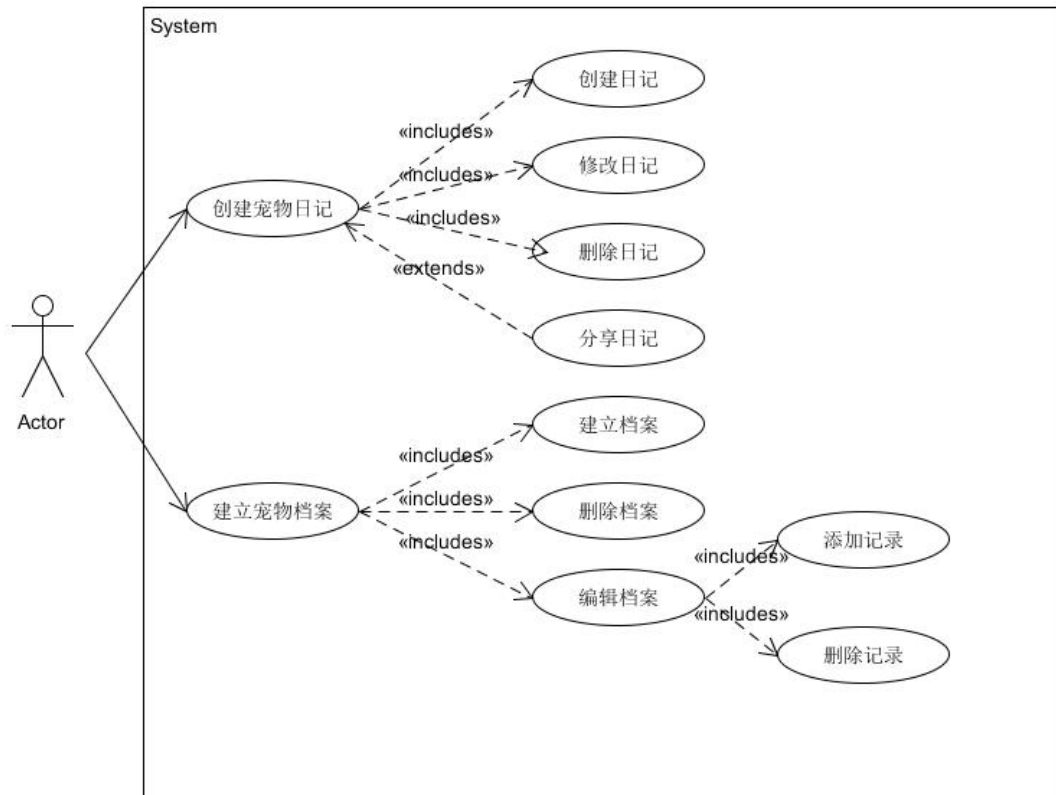


图 3.1 系统用例图

第四章 系统设计

4.1 应用架构设计

本次程序设计基于 MVVM 模式，使用 Android Jetpack 的架构组件并将它们通过图 4.1 的形式构建。

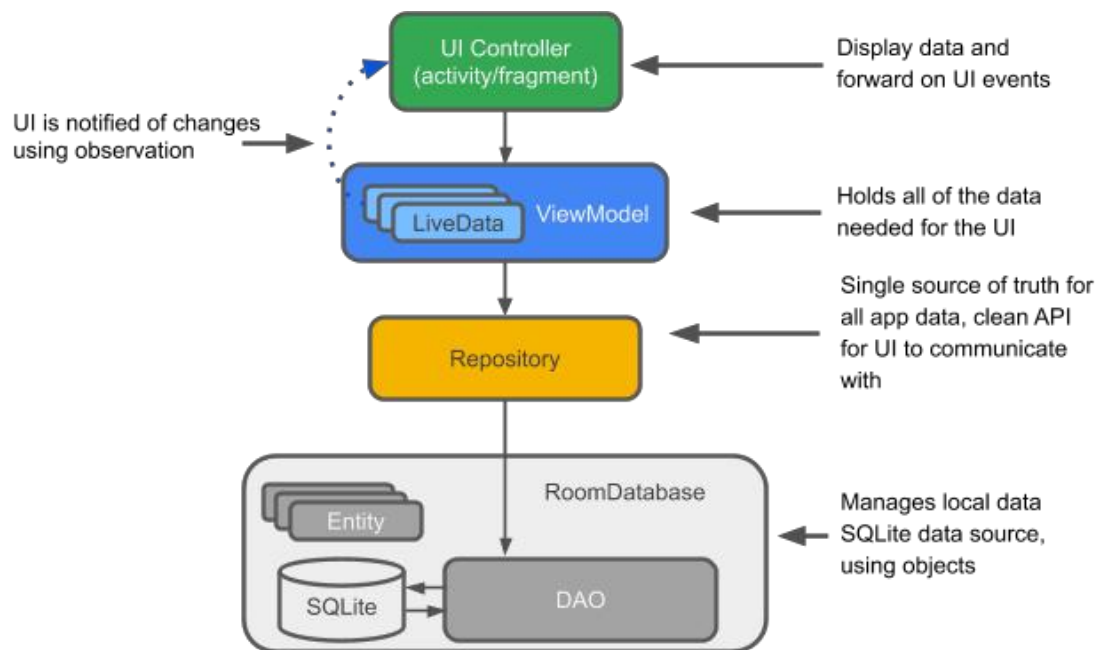


图 4.1 Android Jetpack 推荐应用架构 (图片来源: Google Codelabs.Android Room with a View)

4.1.1 MVVM

MVVM 是一种增强分离关注点的软件架构模式，它将应用程序的图形用户界面与业务逻辑（或后端逻辑）完全分离。

MVVM 模式包含三个核心组件：Model、ViewModel、View，它们之间的关系如图 4.2 显示。

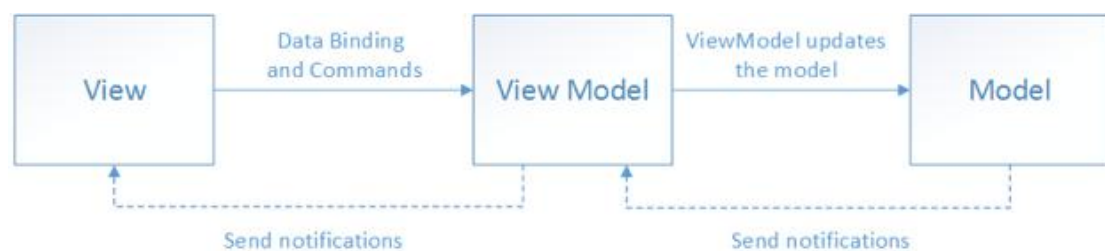


图 4.2 MVVM (图片来源: Microsoft Docs.The Model-View-ViewModel Pattern)

View 封装了 UI 与 UI 逻辑，它负责定义用户在屏幕上看到的结构、布局 and

外观。

ViewModel 封装了表示逻辑和状态，它实现 View 与数据绑定的属性和命令，并可以通过发送通知事件将任何状态通知给 View；除此之外，ViewModel 还负责协调 View 与 Model 类的交互，ViewModel 能够以 View 易于使用的形式提供来自 Model 的数据，必要时 ViewModel 可以进行数据转换。

Model 封装了应用程序数据的非可视类，它通常与封装数据访问和缓存的服务或存储库结合使用。

使用 MVVM 模式具有以下优点：

(1) 对一个已经实现了的封装了业务逻辑的 Mode 类进行修改存在困难以及可能带来风险，在这种情况下，ViewModel 充当 Model 类的适配器并且可以帮助开发者避免 Model 类代码的修改。

(2) 开发者可以在不使用 View 的情况下对 ViewModel 与 Model 创建单元测试。

(3) 如果一个应用的 View 全部在诸如 XML 的资源文件中实现，则开发者可以不需要修改 ViewModel 而重新建立应用的 View。

(4) 设计人员与开发者可以独立分开来工作，设计人员仅需要关注 View，而开发者则专注 ViewModel 与 Model。

4.1.2 系统架构设计

“喵呜”APP 的系统架构如图 4.3 所示。

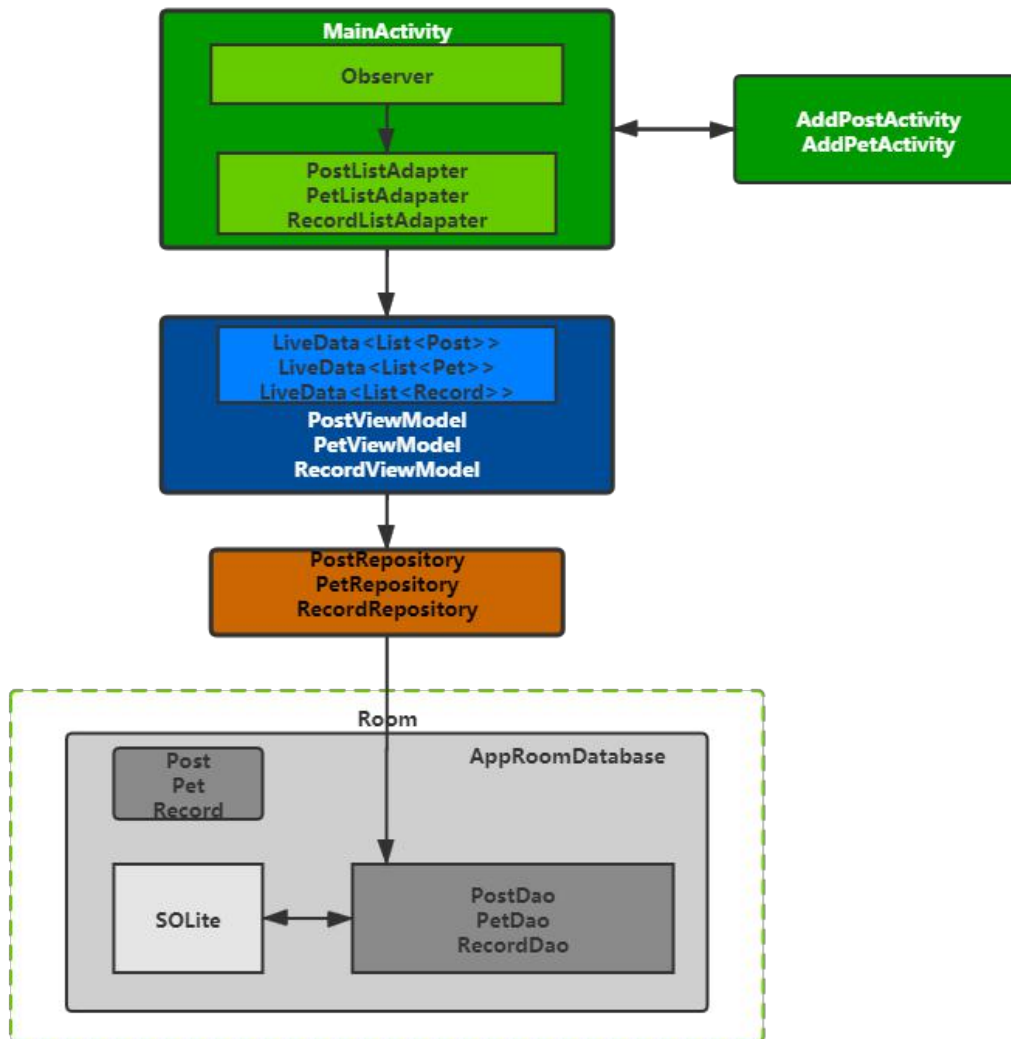


图 4.3 APP 系统架构图

下面对图 4.3 内容进行说明:

图 4.3 中除 SQLite 之外的各个闭合框均为 APP 中创建的类。

- **Post**、**Pet**、**Record** 均为实体类，它们分别表示宠物日记实体、宠物档案实体、宠物健康记录实体，它们描述使用 Room 时的数据库表。
- **SQLite** 是 Room 的底层数据库，Room 持久性库为开发者创建并维护数据库。
- **PostDao**、**PetDao**、**RecordDao** 均为 DAO 类，它提供了将用户数据更新到数据库的方法。
- **AppRoomDatabase** 为 APP 的数据库类，它是一个抽象类，包含 **Post**、**Pet**、**RecordList** 三个数据表。
- **PostRepository**、**PetRepository**、**RecordRepository** 为数据获取过程委派的一个

新模块，它们处理相关的数据操作，并提供一个纯净的 API，使得应用的其余部分可以轻松检索数据。Repository 类可以处理从何处获取数据以及调用哪些 API。对于一个可能同时包含从后端获取数据与持久性库的 APP，Repository 类可以充当持久性模型、网络服务与缓存这样不同数据源之间的媒介。

- PostViewModel、PetViewModel、RecordViewModel 为相关的界面组件提供数据，并包含数据处理业务逻辑。
- LiveData<List<Post>>、LiveData<List<Pet>>、LiveData<List<Record>> 为可观察的数据存储器，更新数据时，系统会通知相关的 Activity 或 Fragment。
- Adapter 为适配器，它负责创建 ViewHolder，并将 ViewHolder 绑定到相应的数据。
- MainActivity 是一个抽屉式导航栏，它在表示宠物日记与宠物档案的 PostFragment、ArchiveFragment 之间进行导航任务；PostFragment、ArchiveFragment 使用 RecyclerView 进行界面布局，它们通过一个悬浮按钮菜单跳转到 AddPostActivity 和 AddPetActivity，AddPostActivity 与 AddPetActivity 将列表项添加到对应的 RecyclerViewList。

4.2 数据库设计

4.2.1 数据库概念设计

通过系统分析，可以得到本应用程序中总计有三个表：宠物日记表、宠物档案表以及宠物疾病诊断、用药历史等的记录表。其中宠物日记表与宠物档案表之间独立，宠物档案表与宠物记录表之间为一对多的关系。图 4.4 显示了系统 E-R 图。

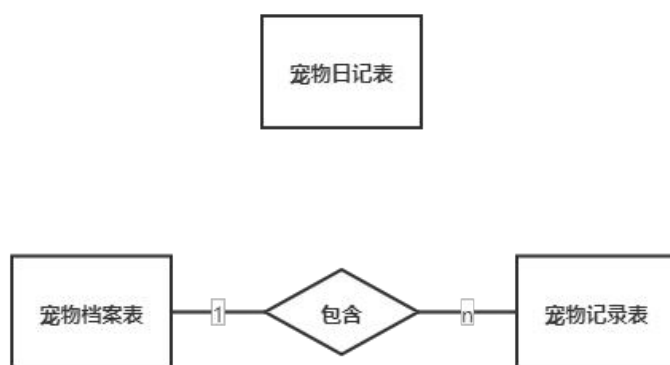


图 4.4 系统 E-R 图

4.2.2 数据表设计

下面对本程序中的三个数据表所包含的字段以及字段大小进行说明。

(1) 宠物日记表

宠物日记表存储用户发布的日记信息，其应该包含用户发布的日记的各项属性，在分析了各字段的基础作用之后得到的数据表如表 1 所示。

表 1：宠物日记表

字段名	类型	描述
postId	int	日记标识，主键
postTime	String	日记发布的时间
postText	String	日记文字内容
postImageUrl	String	日记照片的路径

(2) 宠物档案表

宠物档案表目的是记录关于宠物的所有健康信息，该表应该包含宠物基本信息以及健康相关的记录，其中宠物健康信息存储为另一个数据表，这里通过外键与其他表关联。宠物档案数据表如表 2 表示。

表 2：宠物档案表

字段名	类型	描述
petName	String	宠物名称，主键
petGender	String	宠物性别
PetBirth	String	宠物出生日期
petProfileUrl	String	宠物头像图片的路径
petRecordId	int	与宠物记录表关联，外键

(3) 宠物记录表

宠物记录表则包含了宠物的疾病诊断记录、药品使用记录、疫苗注射记录、针对猫犬宠物的驱虫记录等, 宠物档案表与该表为一对多的关系, 通过外键关联。宠物记录表各字段信息如表 3 所示。

表 3: 宠物记录表

字段名	类型	描述
recordId	int	记录标识, 主键
postList	ArrayList<>	关于宠物的所有健康记录

第五章 系统实现与测试

5.1 系统实现

5.1.1 开发环境

本 APP 的实现基于 Mac OS 平台进行构建，使用 Android Studio 平台和 Android SDK 组件加以辅助。

以下数据为 APP 开发环境详细参数：

Android Studio 3.6.1

Build #AI-192.7142.36.36.6241897, built on February 27, 2020

Runtime version: 1.8.0_212-release-1586-b4-5784211 x86_64

VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

macOS 10.15.4

GC: ParNew, ConcurrentMarkSweep

Memory: 1981M

Cores: 8

Registry: ide.new.welcome.screen.force=true

Non-Bundled Plugins:

5.1.2 功能模块实现

(1) APP 目录结构

图 5.1 与图 5.2 分别展示了 APP 的 java 目录结构与 res 目录结构：



图 5.1 APP java 目录结构

图 5.2 APP res 目录结构

(2) 主页面实现

主页面 MainActivity 是一个抽屉式导航栏，默认显示界面是宠物日记界面 PostFragment。图 5.3 显示了主页面的导航栏。

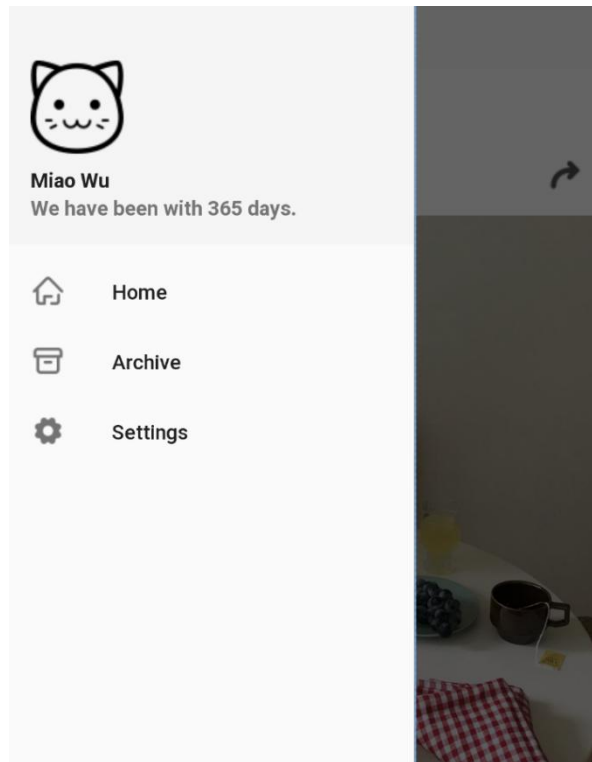


图 5.3 抽屉式导航栏

(3) 宠物日记界面

该界面为 `PostFragment`，使用 `RecyclerView` 创建列表，点击列表项可以重新编辑日记，每个列表项中设置了分享与删除按钮。图 5.4 展示了该界面。

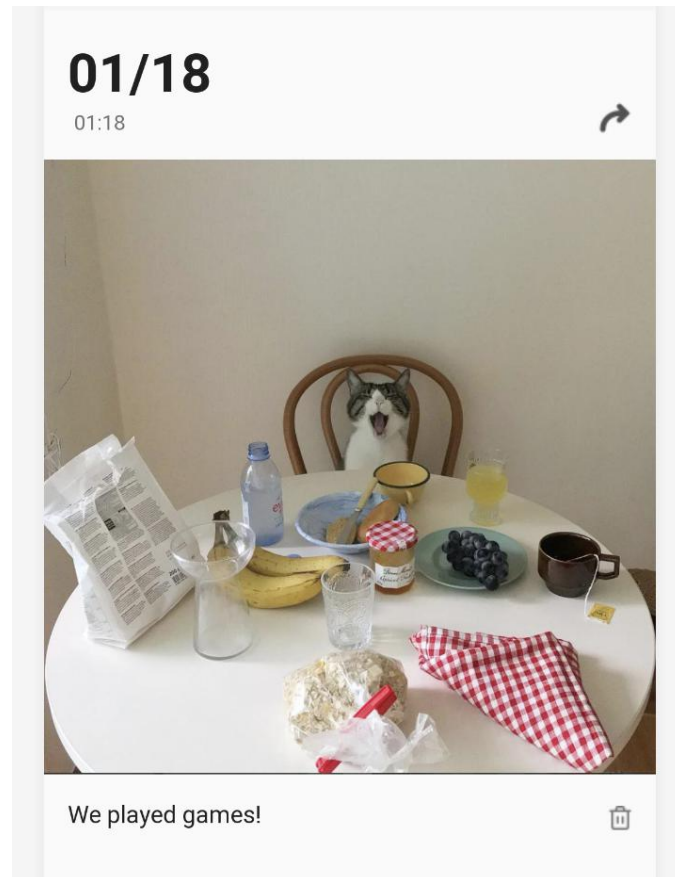


图 5.4 宠物日记

(4) 创建宠物日记界面

该界面为 `AddPostActivity`，通过主页面的 `FloatActionButton` 可以跳转到该页面，用户可以取消编辑日记、完成日记、打开相机或者图库添加照片。图 5.5 展示该界面。

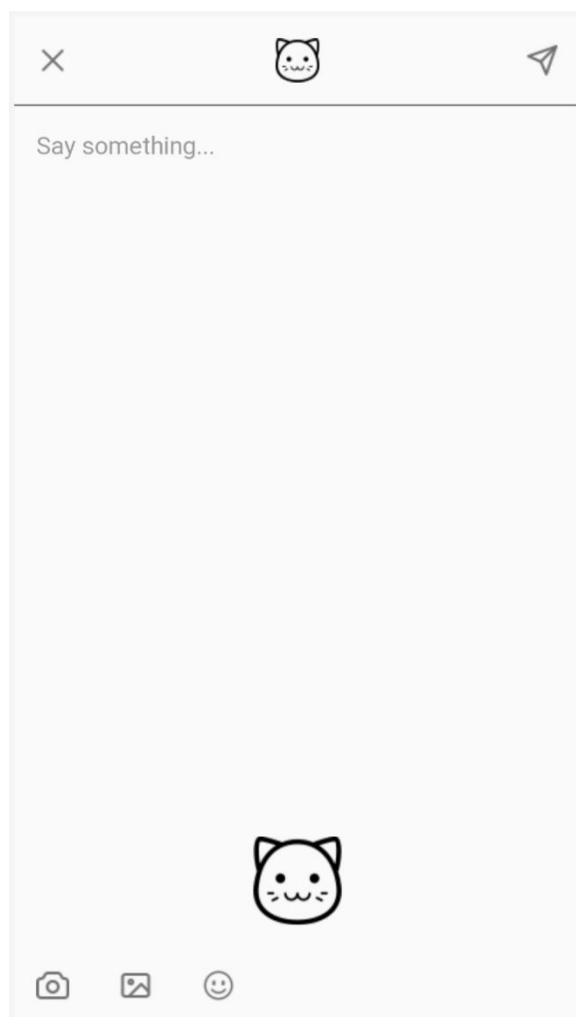


图 5.5 创建宠物日记

(5) 宠物档案界面

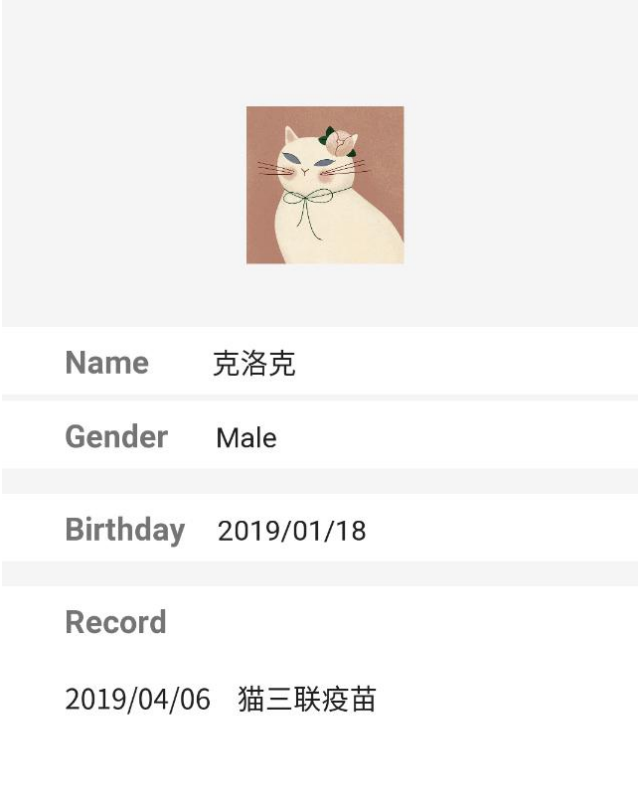
该界面为 ArchiveFragment，使用 RecyclerView 创建了一个 Grid 布局，点击列表项可以查看与编辑宠物档案详情。图 5.6 展示该界面。



图 5.6 宠物档案界面

(6) 编辑宠物档案界面

该界面为 AddPetActivity，用户可以在该界面查看到宠物档案详情，并在该页面添加宠物健康信息记录。图 5.7 展示该界面。



Name	克洛克
Gender	Male
Birthday	2019/01/18
Record	
2019/04/06	猫三联疫苗

图 5.7 编辑宠物档案

5.2 系统测试

系统测试是软件生命周期中的重要一环，通过系统测试可以及时找出系统中存在的缺陷，并针对缺陷进行处理。在程序开发过程中，主要测试程序代码是否存在错误、功能是否满足需求、系统运行是否存在缺陷等。

5.2.1 功能测试

功能测试主要包括：

(1) 界面测试：主要对界面布局、显示效果、用户体验、界面元件标识等进行检测。

(2) 必填项测试：在用户进行输入时，检查输入格式是否正确以及是否有提示信息等。

(3) 检查按钮测试: 检查界面中的各个按钮能否响应用户操作, 能否正确链接页面等。

(4) 页面链接测试: 测试页面之间能否正确跳转, 页面返回操作的正确性等。

5.2.2 测试结果分析

通过进行系统测试以及对系统缺陷进行修改之后, 应用程序代码无错误, 系统功能满足需求分析与设计, 系统能够正确运行。

第六章 总结与展望

6.1 总结

本文使用 Android Jetpack 工具库的 LiveData、Navigation、Room 等组件实现了一个本地数据存储的宠物类 APP。

该 APP 摒弃了传统宠物类 APP“社交+电商”的定位模式, APP 的分析与设计重新关注宠物本身, 关注人与宠物的关系。

在 APP 技术开发方面综合使用本科学习的 Android 开发技术, 并在此基础上进行了新的尝试与提升:

(1) 在数据存储器选择上尝试使用具有生命周期感知能力的 LiveData, 降低了程序代码的复杂度, 使得代码更容易维护;

(2) 使用 Android Jetpack 工具库与其 Navigation 组件, 处理应用内导航所需的一切工作。

(3) 放弃直接使用 SQLite 数据库, 通过 Room 持久性库将应用数据进行本地存储, 简化了模板代码, 程序更易编写。

本次程序设计也是一次比较完整的软件开发过程, 包括了软件生命周期中的需求分析阶段、软件设计阶段、软件测试阶段。

6.2 展望

尽管“喵呜”APP 的产品定位为宠物本身, 但就目前而言该 APP 的功能仍然较为简单与不足, 并不能覆盖宠物的所有需求, 在 APP 的后续版本中将添加宠物百科、宠物养护指南、宠物在线诊断等更多关注宠物健康的功能。

除此之外, 该 APP 是一个单用户体系结构的 Android 应用, 当用户执行手机清理操作后 APP 的数据极易丢失, 因此该 APP 的下一步发展方向将采用 C/S (Client/Server) 系统体系结构, 实现用户数据在服务端的存储与访问。

C/S 结构即客户端与服务端结构, 这种体系结构是在信息系统软件支持下的两层结构模型, 它以数据库服务器为中心, 以客户端为网络基础。在 C/S 体系结构中, 数据存储在服务端的数据库上, 用户操作在客户端执行。客户端通过服务端获取数据与网络资源。

“喵呜”APP 未来将以 C/S 体系结构为基础，将数据库部署在服务端，同时在客户端依然使用 Room 持久性库进行数据缓存，使得用户在离线情况下仍然可以获取数据，当设备连接到网络后，将数据更改同步到服务端。

参考文献

- [1]. 刘靖宇,徐志超.智能手机应用系统的现状及发展趋势[J].绿色科技,2019,(18):208-209.
- [2]. 张瑞春,王金合,郑宝亮.国内宠物医疗行业人才发展现状探析[J].畜牧与饲料科学,2019,40(07):73-76.
- [3]. Android Developer.Android Jetpack[J].<https://developer.android.com/jetpack>, 2020.3
- [4]. Android Developer.LiveData Overview[J].<https://developer.android.com/topic/libraries/architecture/livedata>,2020.3
- [5]. Android Developer.Navigation[J].<https://developer.android.com/topic/libraries/architecture/navigation>,2020.3
- [6]. Android Developer.Save data in a local database using Room[J].<https://developer.android.com/training/data-storage/room>,2020.3
- [7]. Microsoft Docs.The Model-View-ViewModel Pattern[J].<https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm>,2020.3
- [8]. Google Codelabs.Android Room with a View[J].<https://codelabs.developers.google.com/codelabs/android-room-with-a-view/#0>,2020.3
- [9]. 欧运娟.基于 Android 的在线学习系统的设计与实现[J].计算机产品与流通, 2020(03):127-128.
- [10]. 王新宇.基于 Android 系统的移动学习平台的设计与实现[J].电脑知识与技术. 2015(19):76-79.

致 谢

从本次论文开题到完成的过程中，首先感谢我的指导老师刘聪老师，在他的指导与监督下，我才能够有质量地完成本次毕业设计。

另外，我感谢本科四年所有教授我课程的老师，本次毕业设计是以他们传授的知识为基础而完成的。

我也感谢在本次论文完成过程中所有给予我帮助的家人、老师、同学、朋友，他们给了我精神支持并对我的应用程序提出了可行性建议。

最后，感谢 Android Developer 提供的文档支持。

附录

附录 A

附录 A 为本程序中 Navigation 组件的程序代码。

- **mobile_navigation.xml:**

图附录 A1 展示了 mobile_navigation.xml 文件代码:

```
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_home">

    <fragment
        android:id="@+id/nav_home"
        android:name="com.chenyn.miaowu.ui.home.HomeFragment"
        android:label="@string/menu_home"
        tools:layout="@layout/fragment_home">

        <action
            android:id="@+id/action_HomeFragment_to_HomeSecondFragment"
            app:destination="@id/nav_home_second" />
        </fragment>
    <fragment
        android:id="@+id/nav_home_second"
        android:name="com.chenyn.miaowu.ui.home.HomeSecondFragment"
        android:label="@string/home_second"
        tools:layout="@layout/fragment_home_second">
        <action
            android:id="@+id/action_HomeSecondFragment_to_HomeFragment"
            app:destination="@id/nav_home" />

        <argument
            android:name="myArg"
            app:argType="string" />
        </fragment>

    <fragment
        android:id="@+id/nav_archive"
        android:name="com.chenyn.miaowu.ui.archive.ArchiveFragment"
        android:label="@string/menu_archive"
        tools:layout="@layout/fragment_archive" />

    <fragment
        android:id="@+id/nav_settings"
        android:name="com.chenyn.miaowu.ui.settings.SettingsFragment"
        android:label="@string/menu_settings"
        tools:layout="@layout/fragment_settings" />
</navigation>
```

图附录 A1 mobile_navigation.xml

- **activity_main_drawer.xml:**

图附录 A2 展示了 activity_main_drawer.xml 文件代码:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_home"
            android:icon="@drawable/home"
            android:title="@string/menu_home" />
        <item
            android:id="@+id/nav_archive"
            android:icon="@drawable/archive"
            android:title="@string/menu_archive" />
        <item
            android:id="@+id/nav_settings"
            android:icon="@drawable/settings"
            android:title="@string/menu_settings" />
    </group>
</menu>
```

图附录 A2 activity_main_drawer.xml

- **MainActivity 中导航组件相关代码:**

图附录 A3 展示了 MainActivity 中导航组件相关代码:

```
DrawerLayout drawer = findViewById(R.id.drawer_layout);
NavigationView navigationView = findViewById(R.id.nav_view);
// Passing each menu ID as a set of Ids because each
// menu should be considered as top level destinations.
mAppBarConfiguration = new AppBarConfiguration.Builder(
    R.id.nav_home, R.id.nav_archive, R.id.nav_settings)
    .setDrawerLayout(drawer)
    .build();
NavController navController = Navigation.findNavController(this,
    R.id.nav_host_fragment);
NavigationUI.setupActionBarWithNavController(this, navController,
    mAppBarConfiguration);
NavigationUI.setupWithNavController(navigationView, navController);
```

图附录 A3 MainActivity 中导航组件相关代码

附录 B

附录 B 为 Room 持久性库相关代码，以宠物日记 Post 对象为例。

- **Post 实体类:**

图附录 B1 展示了 Post 实体类的代码:

```
@Entity(tableName = "post_table")
public class Post {

    @PrimaryKey(autoGenerate = true)
    @NonNull
    @ColumnInfo(name = "post_id")
    private int postId;

    @ColumnInfo(name = "post_time")
    private String postTime;

    @ColumnInfo(name = "post_text")
    private String postText;

    @ColumnInfo(name = "post_image_url")
    private String postImageUrl;

    public Post(String postTime, String postText, String postImageUrl) {
        this.postTime = postTime;
        this.postText = postText;
        this.postImageUrl = postImageUrl;
    }

    //setter, getter...
}
```

图附录 B1 Post 实体类

- **PostDao:**

图附录 B2 展示了 PostDao 类的代码:


```

@Dao
public interface PostDao {
    @Insert
    void insertPost(Post post);

    @Delete()
    void deletePost(Post post);

    @Query("DELETE FROM post_table WHERE post_id = :postId")
    void deletePostById(int postId);

    @Query("DELETE FROM post_table")
    void deleteAllPosts();

    @Query("SELECT * FROM post_table")
    LiveData<List<Post>> getAllPosts();
}

```

图附录 B2 PostDao 类

- **PostRepository:**

图附录 B3 展示了 PostRepository 类的代码:

```

public class PostRepository {
    private PostDao mPostDao;
    private LiveData<List<Post>> mAllPosts;

    public PostRepository(Application application) {
        AppRoomDatabase db = AppRoomDatabase.getRoomDatabase(application);
        this.mPostDao = db.postDao();
        this.mAllPosts = mPostDao.getAllPosts();
    }

    public LiveData<List<Post>> getAllPosts() {
        return mAllPosts;
    }

    public void insertPost(final Post post) {
        AppRoomDatabase.databaseWriteExecutor.execute(() -> {
            mPostDao.insertPost(post);
        });
    }

    public void deletePostById(int postId) {
        AppRoomDatabase.databaseWriteExecutor.execute(() -> {
            mPostDao.deletePostById(postId);
        });
    }
}

```

图附录 B3 PostRepository 类

- **AppRoomDatabase:**

图附录 B4 展示了 AppRoomDatabase 类的代码:

```

@Database(entities = {Post.class}, version = 1, exportSchema = false)
public abstract class AppRoomDatabase extends RoomDatabase {

    public abstract PostDao postDao();

    // marking the instance as volatile to ensure atomic access to the variable
    private static volatile AppRoomDatabase INSTANCE;
    private static final int NUMBER_OF_THREADS = 4;
    public static final ExecutorService databaseWriteExecutor =
        Executors.newFixedThreadPool(NUMBER_OF_THREADS);

    private static final String DB_NAME = "RoomDatabase";

    public static AppRoomDatabase getRoomDatabase(Context context) {
        if (INSTANCE == null) {
            synchronized (AppRoomDatabase.class) {
                if (INSTANCE == null) {
                    INSTANCE =
Room.databaseBuilder(context.getApplicationContext(),
                        AppRoomDatabase.class, DB_NAME)
                            .addCallback(sRoomDatabaseCallback)
                            .build();
                }
            }
        }
        return INSTANCE;
    }

    private static RoomDatabase.Callback sRoomDatabaseCallback = new
RoomDatabase.Callback() {
        @Override
        public void onOpen(@NonNull SupportSQLiteDatabase db) {
            super.onOpen(db);

            databaseWriteExecutor.execute(() -> {
                PostDao dao = INSTANCE.postDao();
                dao.deleteAllPosts();

                Post post = new Post("pre_postTime", "pre_postText",
"pre_postImageUrl");
                dao.insertPost(post);
            });
        }
    };
}

```

图附录 B4 AppRoomDatabase 类

毕业论文（设计）成绩评定记录

Grading Sheet of the Graduation Thesis (Design)

指导教师评语 Comments of Supervisor:

项目设计了一个关注宠物健康的 Android 平台的应用程序，锻炼学生的软件设计和开发能力。该软件具有一定的实用价值，和一定的新颖性（在开题的时候找不到类似软件的存在）。软件开发难度适中。论文结构合理描述清晰。在论文完成过程中学习态度认真，能够在规定时间内完成毕业设计。

成绩评定 Grade: 良好

指导教师签名 Supervisor Signature: 刘聪

Date:

答辩小组意见 Comments of the Defense Committee:

成绩评定 Grade:

签名 Signatures of Committee Members:

Date:

院系负责人意见 Comments of the Academic Chief of School:

成绩评定 Grade:

签名 Signature:

院系盖章 Stamp:

Date:



中山大學
SUN YAT-SEN UNIVERSITY