长沙理工大学

CHANGSHA UNIVERSITY OF SCIENCE & TECHNOLOGY

毕业设计（论文）

题目：基于安卓的日程管理系统

学生姓名：彭国豪

学　　号：201850080441

班　　级：计算机 18-4 班

专　　业：计算机科学与技术

指导教师：阮昌，王盛（企业导师）

2022 年 6 月

基于安卓的日程管理系统

学生姓名：彭国豪

学　　号：201850080441

班　　级：计算机 18-4 班

所在学院：计算机与通信工程学院

指导教师：阮昌，王盛（企业导师）

完成日期：2022 年 6 月

基于安卓的日程管理系统设计与实现

摘要

在科技创新之手的推动下，社会发展的车轮滚滚前行，人们生活质量得到不断提高的同时，每天需要处理的事务也与日俱增，如何按时、高效地完成生活中接踵而来的各项任务已经称为摆在我们面前的一个巨大挑战。

针对生活节奏不断加快的大环境下用户日益增涨的日程管理需求，本文借助Android平台设计和实现了一款功能丰富的日程管理系统，旨在帮助用户在当下快节奏的社会环境中提高学习和工作的效率，增加生活幸福感。

“日程管理”这一行为的核心在于记录，但如果一个日程管理软件仅提供记事本功能是远远无法满足当下用户需求的。记录、查看、提醒这三大功能是用户对日程管理的基本诉求，因此一个合格的日程管理软件只有集记事本、日历、闹钟三者为一体才能满足人们日常生活中绝大多数的使用场景。以此为蓝本，本文所设计和实现的日程管理系统提供了日程收集箱、TODO日历、到点提醒等功能以满足用户的日常使用。而在此基础之上，本系统还提供了定时推送、数据反馈、专注计时等功能以帮助用户合理地规划、利用时间。

关键词：日程管理；Android；效率；TODO

DESIGN AND IMPLEMENTATION OF SCHEDULE MANAGERMENT SYSTEM BASED ON ANDROID

ABSTRACT

Driven by the hand of technological innovation, the wheel of social development is rolling forward. While the quality of people's life is constantly improving, the daily affairs that need to be dealt with are also increasing day by day. How to complete the tasks that follow in life on time and efficiently has been called a huge challenge before us.

Aiming at the increasing schedule management needs of users in the environment of the accelerating pace of life, this paper designs and implements a feature-rich schedule management system with the help of the Android platform, aiming to help users improve learning and learning in the current fast-paced social environment. Work efficiency and increase happiness in life.

The core of schedule management is recording, but if a schedule management software only provides a notepad function, it is far from meeting the needs of current users. The three functions of recording, viewing, and reminding are the basic demands of users for schedule management. Therefore, a qualified schedule management software can only meet the vast majority of usage scenarios in people's daily life only by integrating notepad, calendar, and alarm clock. Based on this, the schedule management system designed and implemented in this paper provides functions such as task collection box, TODO calendar, and arrival reminder to meet users' daily use. On this basis, the system also provides functions such as regular push, data feedback, and focused timing to help users plan and utilize time reasonably.

Key words: Schedule management; Android; Efficiency; TODO

目录

1 绪论

1.1 课题背景及意义

近年来，在无数科研人员的努力下，移动互联网技术的研究取得了突破性进展，世界也逐步迈向了以移动互联网技术为基石的互联互通之路。而作为移动互联网技术的主要产品之一，智能手机愈发受到人们的欢迎，其购买量和使用率也逐年增长，慢慢成为人们的生活必需品之一。据中商情报网最新数据显示，2021 年全球智能手机出货量达到 13.5 亿台，相较于 2020 年的 12.7 亿台，同比增长 7%，但仍未恢复到疫情前 2019 年的水平[1]。与此同时，应用于智能手机的操作系统也层出不穷，包括 Android、iOS、Windows Phone、BlackBerry OS、Symbian 等，堪称百花齐放。凭借着高度开源的特性，Android 系统在市场占有率方面的数据遥遥领先于 iOS 等其他系统，多年来一直稳居智能手机操作系统市场的头把交椅。从 2021 年中国网民智能手机操作系统使用比例来看，Android 系统毫无意外地仍占据智能手机操作系统市场的半壁江山[2]。粉丝众多的 iOS 系统所占领的市场份额远低于 Android，仅占 33.1%。除这两巨头之外的其他系统，如 Windows Phone、BlackBerry OS、Symbian 等，其使用比例分别为 8.1%、5.3%、2.3%。

随着社会逐步向前发展，人们的生活质量在不断提高，但是与此同时人们的生活节奏也越来越快。生活节奏的不断加快意味着当前社会环境中的人们每天需要处理的事务也变得更加繁杂，经常会面临当前的事情还没处理完，新的任务就已经来到的情况。面对接踵而来的任务，我们常常即使是忙得焦头烂额也处理不完，其中很大一部分原因就是需要完成的任务太多以至于我们不知道应该先处理哪个才好，另一部分原因则是生活中往往会有太多的外来因素干扰我们，使我们无法专注于当前事项，导致学习、工作效率低下。因此即使社会和科技的发展使得人们的生活质量得到了很大的飞跃，生活幸福感却不见得有太多的提高。

由此观之，在当前快节奏的社会环境下，人们迫切地需要一些辅助手段或者工具来帮助自己统筹安排生活中的大小事项，合理规划时间、提高工作和学习的效率。因

此，日程管理系统软件随之应运而生。日程管理系统诞生的目的是帮助用户更好地管理时间，在有限的时间内高质量地完成更多的事。换而言之，研究日程管理系统对帮助用户提高生活效率，增加生活幸福感有很大的意义。

1.2 国内外研究现状

随着移动互联网技术的发展和智能手机的普及，目前国内外手机应用市场上的日程管理软件如雨后春笋般冒出，包括 365 日历、滴答清单、WunderList、高效 ToDo、Any.Do、GTasks 和 CalenGo 等产品。这些产品前赴后继地推出，填补了市面上日程管理类软件的空白，也便利了人们的日常生活。

虽然目前市面上存在大量的日程管理系统，但是他们的质量却是良莠不齐的。其中一部分日程管理软件严格来说仅仅是一个增加了记事本功能的日历软件，只具有简单的日程管理功能，它们功能简单，使用场景单一，无法满足用户日常生活中的大多数需求。这类软件只能说是日历软件的扩展，远远谈不上日程管理系统，并且过于单一的功能也得不到用户的青睐，导致它们的下载量也十分低迷。而另一部分比较受用户欢迎的日程管理软件功能显然更为丰富，它们不仅提供了最基础的日历和记事本功能，还提供了日程提醒甚至社交等功能，同时也具有良好的 UI 设计和交互设计，备受用户喜爱。但这类软件往往都有一个功能的弊端：高级功能需要收费甚至软件的下载都需要收费，普通用户仅能使用最基础的日程管理功能，这就导致软件虽然提供了丰富的功能但是大部分用户却无法使用的状况。同时市面上大部分的日程管理软件都没有向用户提供数据的反馈功能，因此用户无法得知某段时间内自己的任务完成状况和工作效率，而这些数据往往是大部分用户都想要看到的。

1.3 论文构成

本文针对基于安卓的日程管理系统的设计与实现展开研究，共分为七个章节，大体结构如下：

（1） 绪论。该章节主要交代了本课题的研究背景及意义、国内外研究现状，并对论文的内容与结构进行了详细的说明。

（2） 开发工具与关键技术介绍。该章节对设计和实现所研究系统的开发工具以及应用到的相关核心技术进行了介绍。

（3） 系统需求分析。该章节对所研究系统进行了需求分析，阐述了系统的功能模块，并介绍和展示了相关业务用例，确定了所研究系统的可行性。

（4） 系统设计。该章节描述了系统的总体设计，规划了具体的功能模块，并说明了相关逻辑。

（5） 系统实现。该章节阐述了系统各模块的具体功能及实现方式。

（6） 系统测试。该章节主要针对所研究系统进行功能测试和兼容性测试，根据测试结果来分析系统的实用价值。

（7） 总结。该章节对全文的核心内容进行概括，总结了系统的实现程度，自己的收获与不足，并对整个过程进行反思。

2 开发工具与关键技术介绍

2.1 开发工具介绍

2.1.1 Android Studio

2013 年 5 月，Google 公司推出了一款全新的 Android 应用集成开发环境——Android Studio。这款功能丰富的产品一经推出就受到了广大 Android 开发者们的推崇与热捧，它迅速取代 Eclipse 成为开发者们的编写 Android 应用程序时的首选开发工具，其诞生具有划时代的意义。

虽然 Android Studio 是 Google 公司以 IntelliJ IDEA 为蓝本进行构建，但是其青出于蓝而胜于蓝。相比于 IntelliJ 或 Eclipse，Android Studio 提供了更为丰富的开发功能，而正是这些功能的出现使得 Android Studio 构建 Android 应用程序的效率得到了极大的提高。Android Studio 新增的功能包括：

(1) 以项目自动化构建开源工具 Gradle 为基础的 Android 应用程序构建系统；

(2) 性能过硬且功能丰富的 Android 设备模拟器；

(3) Apply Changes 功能：无需重启即可将代码和资源的变更推送到正在运行的应用程序上；

(4) 通用代码模板以及 GitHub 集成；

(5) 大量的测试以及捕获性能、可用性、版本兼容性的工具和框架；

(6) C++和 NDK 的支持；

(7) 通过鼠标拖拽组件即可实现 UI 界面设计与调整的布局编辑器；

(8) 支持 ProGuard 和应用签名。

图 2.1 Android Studio

2.1.2 Sourcetree

Sourcetree 是 Atlassian 公司提供的一款可以在 Windows 和 Mac 上使用的免费的 Git GUI。通过 Sourcetree，开发者们可以使用精美简洁的图形化界面来执行各种各样的 Git 操作，而不必再通过命令行执行繁杂的 Git 命令来管理 Git 仓库。脱离繁杂 Git 命令的束缚之后，开发者可以更专注于编码工作，大大地提高了软件开发效率。

图 2.2 Sourcetree

2.2 关键技术介绍

2.2.1 Android

Andy Rubin 是 Android 系统的初代开发者，他在 2003 年 10 月组建团队着手进行 Android 系统的研发。2005 年 8 月，Google 公司很敏锐地发现了 Android 系统的价值，嗅到了其中的商机，于是它低调地收购了 Android 系统及 Andy Rubin 的工作团队，并让 Andy Rubin 继续负责 Android 的研发。2007 年 11 月，Google 公司牵头成立了开发手持设备联盟（Open Handset Alliance），随后该机构开始参与并主导 Android 系统的研发工作。

作为一个基于 Linux 内核的开源操作系统，Android 系统主要服务于智能移动设备，包括智能手机、智能手表、平板电脑等智能产品。据 Statcounter 数据显示，截至 2021 年 4 月，在全球移动操作系统市场中，Google 公司 Android 系统的市场占有率高达 72.2%，Apple 的 iOS 系统市场占有率为 26.99%，其余操作系统的市场占有率之和低于 1%[3]。

图 2.3 Android 概念图

Google 官方将 Android 平台分为五层架构，自上而下分别是：

(1) 应用层(System Apps)：该层负责与用户交互，所有开发人员所编写的应用程序都属于该层，应用程序的丰富多样性使得该层更为灵活和个性化；

(2) 应用框架层(Java API Framework)：该层主要提供一些以 Java 语言编写和实现的 API，Android 开发者们可以发挥他们的创造力，通过这些 API 来调用 Android 系统的丰富的系统组件和服务从而实现各种各样的功能；

(3) 系统运行库层(Native & ART)：Android 系统的许多核心组件和服务都是通过 C 语言和 C++语言实现，该层正是以 C/C++编写实现的原生库，为 Android 系统提供了主要特性支持；

(4) 硬件抽象层(HAL)：该层为存在于 Android 系统的 OS 内核和设备的硬件电路之间、为上层应用提供硬件操作接口的接口层；

(5) Linux 内核层(Linux Kernel)：该层为 Android 系统的基石，为 Android 系统的正常运行提供了核心服务。

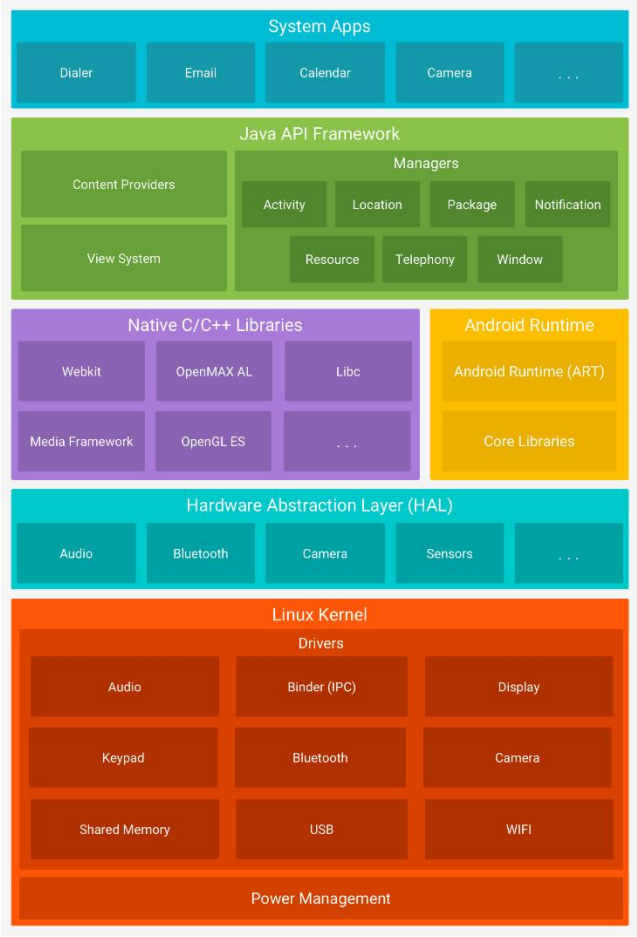

图 2.4 Android 平台架构

2.2.2 RecyclerView

2014 年伴随 Android 5.0 推出的 support-v7 库一经面世就好评如潮，而作为其中的重要组件之一，RecyclerView 自然受到了开发者的热捧，作为一个帮助开发者灵活地展示列表形式或者网格形式的数据（如文本或图片）的容器，它高效替代了 Google 最初推出的 ListView 等列表组件。

在展示列表中的数据项时，实际上只有部分相邻的视图 Item 才会在屏幕上显示。当原本在屏幕之内的视图 Item 因为用户手指的滑动而离开屏幕时，RecyclerView 会复用该视图 Item，将滑入屏幕的新数据填充至其中。正式因为这种通过用回收已有结构取代创建新的列表项的视图复用机制，RecyclerView 才有会比 ListView 具有更高的时间效率和空间效率。

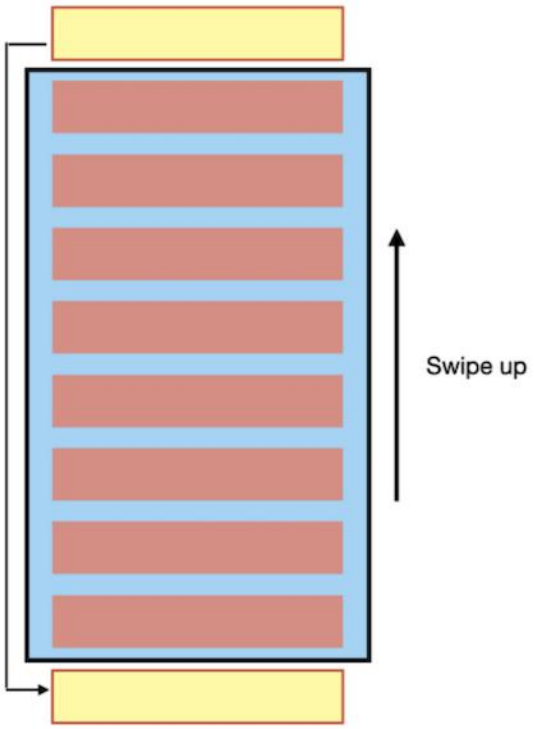

图 2.5 RecyclerView 数据填充原理

RecyclerView 使用灵活、支持个性化定制，具有以下特点：

- RecyclerView 在使用上实现了视图代码解耦，通过 ViewHolder 实现了强大的 Item 复用机制。相较于 ListView，RecyclerView 在访问滑动列表中的视图项时不需要频繁地调用 findViewById()方法，因此它在性能方面的表现更胜一筹。
- RecyclerView 通过 LayoutManager 进行布局管理，LayoutManager 功能强大，

除了系统提供的 LayoutManager，开发者还可以通过继承的方式实现各种各样的布局。

- 在视图动画方面，除了自身默认提供的动画效果，RecyclerView 还允许开发者们创建带有自定义动画效果的滑动列表。

2.2.3 Service

应用程序后台运行能力一直是摆在应用开发者们甚至系统开发者们面前的一道难题，而 Android 交出的答卷则是 Service 组件。Service 作为 Android 系统的四大组件之一，其设计之初的目的就是为了去执行那些需要在后台长期运行的任务。通常来说，Service 在创建时就会和当前应用程序所在的进程绑定，然后一直于该进程中默默地在后台运行。因为相互绑定的关系，如果当前应用程序进程因为系统资源调度或者其他问题被系统终止，Service 也会随之陷入停止状态。值得一提的是，即使当前应用程序被用户切换到了系统后台，Service 也能够正常运行，换而言之，Service 的运行时不受用户界面所束缚的。

至于 Service 和其他系统组件的通信问题，Android 开发者们可以通过 IPC 机制将 Service 和其他系统组件进行绑定，然后两者之间就可以进行交互了，比如可以在其他组件中调用 Service 在系统后台进行网络操作或者与 Content Provider 交换数据。

2.2.4 SQLite

2000 年 8 月，D.RichardHipp 发布了 SQLite 1.0。这是一款开源的轻量级嵌入式关系数据库，在遵循 ACID 的同时，SQLite 还具备运算速度快、占用资源少的特点，是目前主流的适用于智能移动设备的嵌入式数据库。

为了使本地持久化功能的优化取得突破性进展，Android 的研发团队决定在系统中接入 SQLite 这个轻量级数据库，现在看来，他们显然达到了预期的目标。SQLite 在使用上对 Android 开发者们也很友好，除了官方提供的 SQLiteOpenHelper 等帮助类之外，Github 上还存在诸如 LitePal、GreenDao 等许多业界先驱开发的三方数据库操作框架帮助用户使用 SQLite 实现数据的持久化。

3 系统需求分析

3.1 系统功能需求

该系统作为一个基于 Android 平台的日程管理系统，为用户提供的是日常管理方面的服务，包括以下功能：添加日程、删除日程、修改日程、查看日程、日程提醒、数据反馈、番茄计时等功能。其中，以日程的增删改查为系统的核心功能。

3.1.1 添加日程

用户在主界面可以根据日历所选择的日期，通过底部的输入框添加当前日期下待完成的日程。在此过程中用户仅需输入日程标题即可，如果用户需要在指定时间点（某小时）完成日程，同样可以通过底部输入框中的控件完成待做日程精确时间的设置。

3.1.2 删除日程

如果用户需要删除某个日程，可以在日程列表界面长按该日程项，通过弹出的删除对话框实现该需求。同时用户也可以在用户详情界面通过删除按钮实现具体日程的删除。

3.1.3 修改日程

当日程信息（包括标题、内容、时间）发生改变时，用户可以通过点击日程列表页中待修改日程项进入到日程详情页，然后在日程详情页对日程信息进行修改，修改后的日程数据自动保存到本地数据库中。

3.1.4 查看日程

本系统为用户提供了多种不同的视图以查看日程信息。用户可以在主页面通过切

换日期来查看不同日期下的待做事件列表；用户也可以在收集箱页面查看全部未完成和已完成的日程；用户还可以通过点击日程列表中具体的日程项进入日程详情页面，查看该日程的完成情况、时间、标题、内容等详细信息。

3.1.5 日程提醒

本系统可以根据用户在日程项中设置的具体时间（分钟），在指定时间到达时系统会通过通知栏对用户进行提醒；同时系统也会在每天八点在通知栏中将当天待做事项全部推送给用户。用户可以在添加日程时就设置提醒时间，也可以在添加日程时不设置具体时间，后续通过日程详情页进行设置。

3.1.6 数据反馈

系统会根据后台收集到的用户数据，将用户本周以及本月的日程完成情况、专注时长等数据以图表的形式在特定的界面反馈给用户，以供用户参考。

3.1.7 番茄计时

系统根据番茄学习法提供了番茄钟以帮助用户专注于当前事项，提高日程完成的效率。同时后台也会记录相关数据，并以图表的形式实时反馈给用户。

3.2 业务用例分析

根据对功能需求的分析可以得出系统的业务用例如下图：

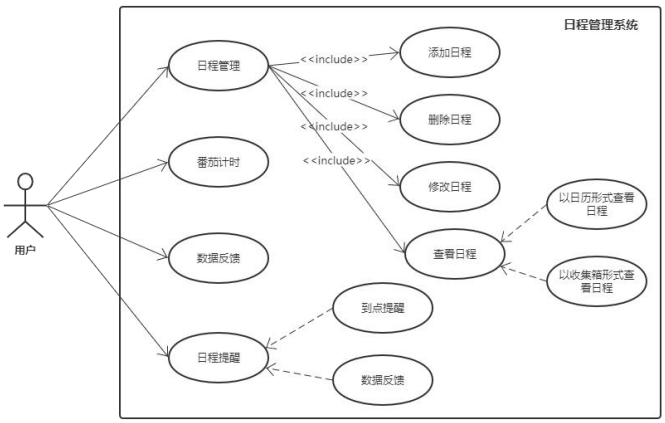

图 3.1 系统业务用例图

4 系统设计

4.1 业务实现方案设计

4.1.1 添加日程

点击主页底部的输入框后会弹出定制的 Dialog 以供用户输入待添加的日程信息。输入完毕后，点击确定按钮，系统会先检查用户的输入是否合法，如果输入合法则会将日程数据保存到 SQLite 中，然后刷新视图，将用户新添加的日程在正确的位置展示。

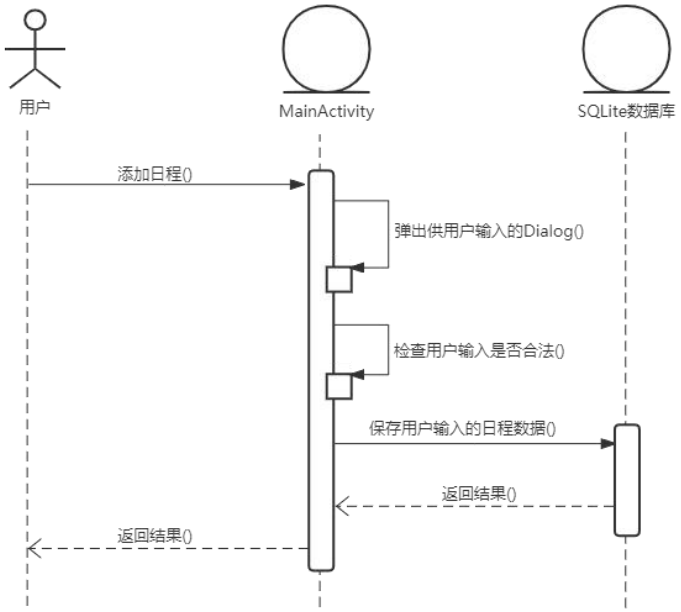

图 4.1 添加日程业务序列图

4.1.2 删除日程

通过长按目标 Item 项可以实现日程的删除。当系统监听到列表中某个日程 Item 被长按时，会弹出删除 Dialog，用户点击确定后，系统会先删除 Adapetr 中缓存的目标日程数据，然后再删除数据库中的目标日程数据。之后系统会根据返回结果刷新视图，

展示最新的数据。

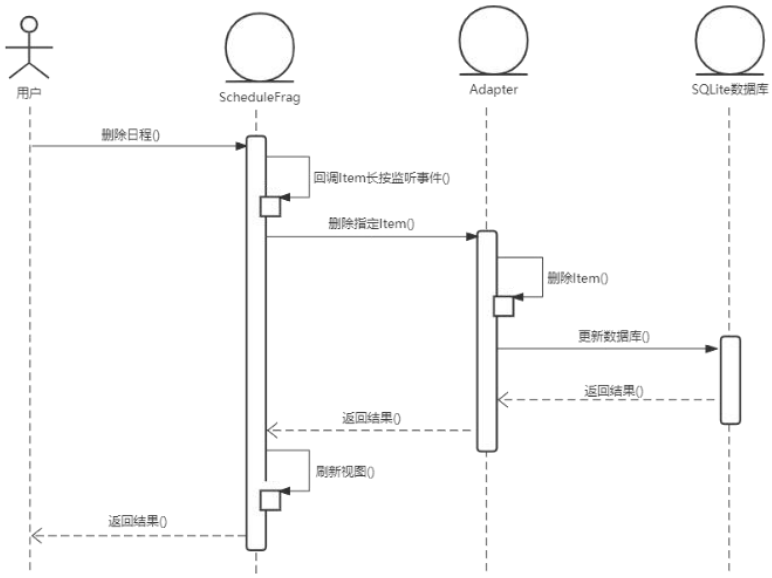

图 4.2 删除日程业务序列图

4.1.3 修改日程

用户如果需要修改现有日程的信息，可以通过点击日程列表中目标日程 Item 的方式进入日程详情页面，该页面的职责是展示和修改日程的具体信息，包括标题、内容、时间、完成状态。在日程详情页修改日程信息后，会同步更新到 SQLite 数据库中。

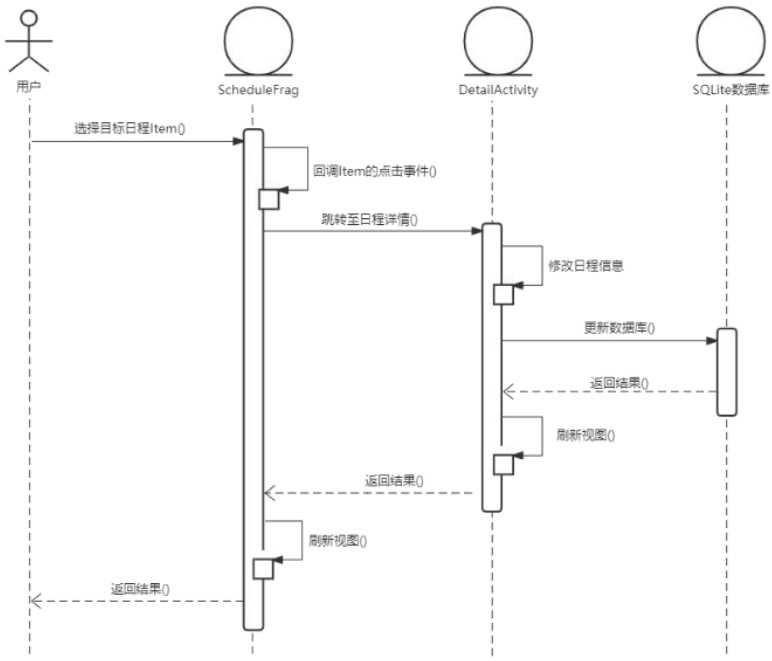

图 4.3 修改日程业务序列图

4.1.4 查看日程

系统启动时，主动从数据库中读取用户的日程数据，然后通过 RecyclerView 控件将日程数据已列表的形式展示。在初始化 RecyclerView 时，需要通过 Adapter 完成数据和 ItemView 的适配，然后才能正常地展示列表数据。

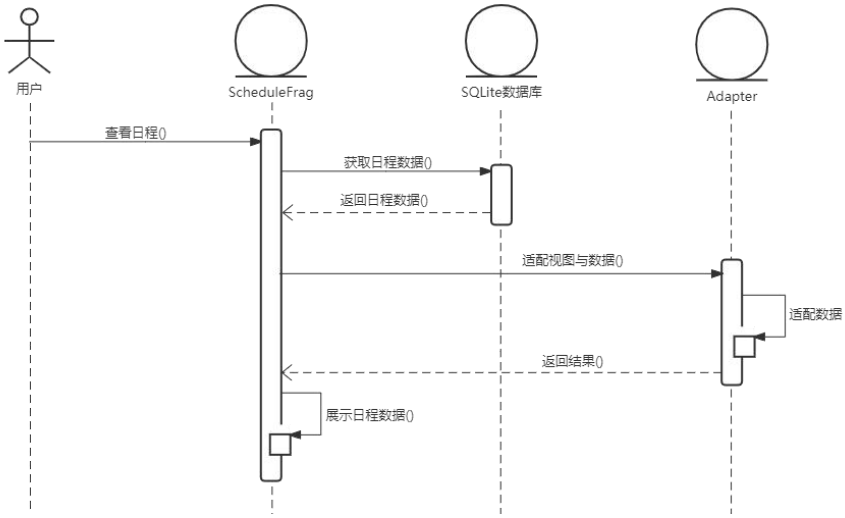

图 4.4 查看日程业务序列图

4.1.5 日程提醒

用户可以通过设置日程的具体时间，让系统在指定的时间点以通知的形式提醒自己完成任务。该功能主要通过 Android 四大组件之一的 Service 实现：用户可以在添加/修改日程时设置日程的具体时间，一旦系统监听到某个日程被设置了具体的完成时间，则会在后台的 Service 中通过 AlarmManager 设置一个目标时间点的闹钟，在指定时间唤醒手机系统发出通知提醒用户完成日程。因为 Servcie 的特殊性质，即使应用程序不在前台运行用户仍然能得到日程提醒

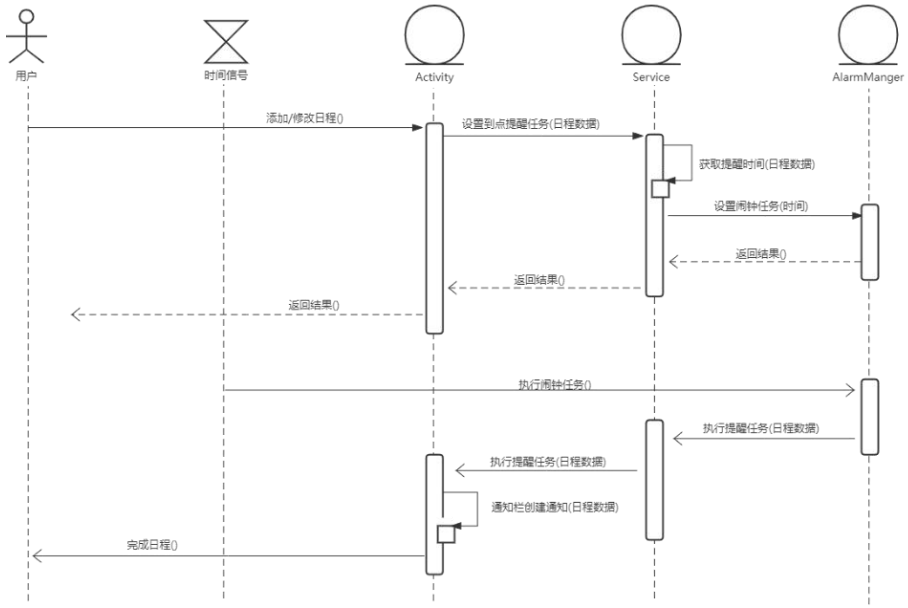

图 4.5 日程提醒业务序列图

4.1.6 数据反馈

在数据统计页面，系统会将用户最近一周、最近一月的日程完成情况、专注时长等数据以折线图、饼状图等形式反馈给用户。该功能借助于第三方框架 HelloCharts 实现：ChartsActivity 启动时，系统会从 SQLite 数据库中读取相关数据，然后对数据进行加工处理以适配 HelloCharts 提供的接口，最后通过 HelloCharts 将相关数据以图表的形式展示给用户。

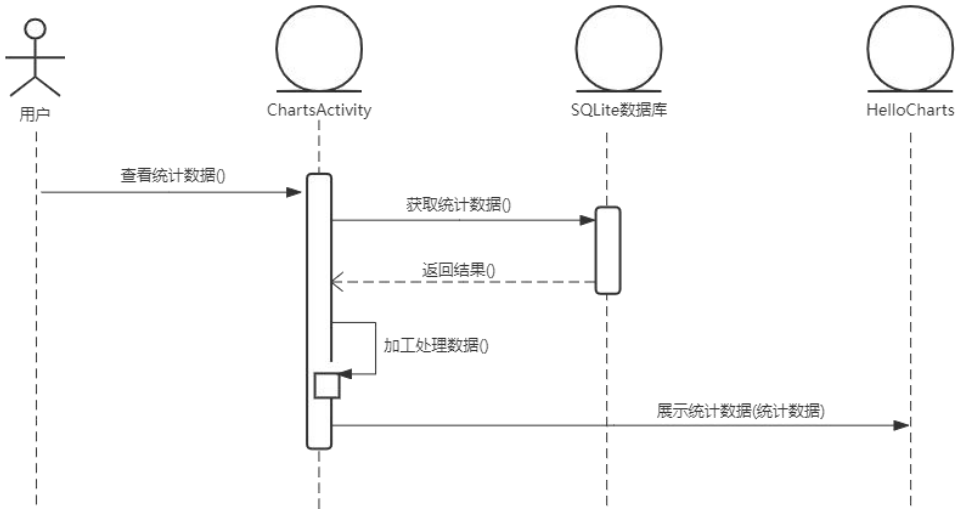

图 4.6 数据反馈业务序列图

4.1.7 番茄计时

在番茄计时页面中，用户可以通过上下滑动手指来设置计时时长，然后点击开始计时即可。番茄计时的目的是根据番茄学习法帮助用户提高对当前事项的专注度，以提高工作效率。该功能的核心是通过自定义 View 的方式实现的，通过自定义 View 将视图绘制和 Android 提供的计时器相结合，从而达到边计时边展示进度的目的。

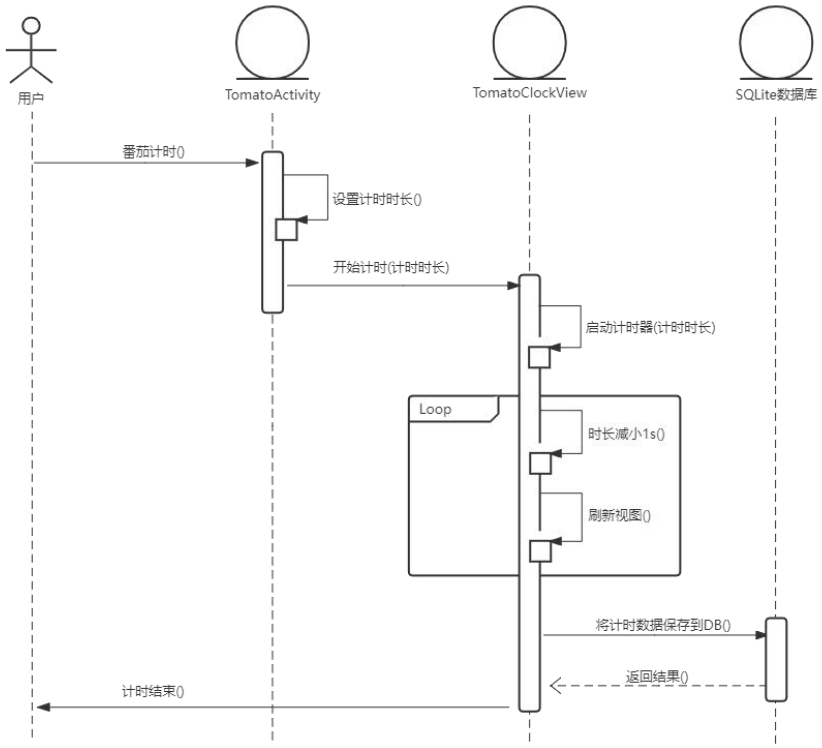

图 4.7 番茄计时业务序列图

4.2 数据库设计

根据基于 Android 的日程管理系统的需求分析，我们设计并实现了 Schedule、ConcentrationData 两种表用以承载系统功能。

4.2.1 日程信息表

日程信息表（Schedule）是系统的核心表，用于记录用户的日程信息，包括日程的

标题、描述、完成状态、计划完成时间、日期等数据项。

表 4- 1 日程信息表

名称	说明	类型	是否为空	是否为主键
id	唯一标识符	int	否	是
title	日常标题	text	否	否
des	日程描述	text	否	否
is_finish	完成状态	int	否	否
time	日程目标完成时间	int	否	否
year	年	int	否	否
month	月	int	否	否
day	日	int	否	否

4.2.2 专注时长表

专注时长表（ConcentrationData）用于记录用户的专注数据，包括专注时长、日期等数据，其中专注时长是当前所记录日期的总专注时长。通过该表记录用户的专注数据，用于实现数据反馈功能。

表 4-2 专注时长表

名称	说明	类型	是否为空	是否为主键
id	唯一标识符	int	否	是
duration	专注时长	int	否	否
year	年	int	否	否
month	月	int	否	否
day	日	int	否	否

5 系统实现

5.1 主界面实现

系统主界面分为三部分实现：顶部导航栏、中间主体 FrameLayout、底部输入框控件。顶部导航栏采用 Toolbar 实现，左侧是一个菜单按钮，用于弹出 DrawLayout 的菜单，用户可以通过此菜单切换界面使用不同的功能，中间位置是两个 TextView，用以展示用户当前选中日期数据；中间主体使用 FrameLayout 承载 ScheduleFrag 和 EventSetFrag 的 Fragment，其中 ScheduleFrag 为主界面的展示控件，用以展示日历视角下的日程列表，通过点击日历上不同的日期，可以实现日期的切换，系统会随之展示不同日期下的 TODO List（即日程列表），日程列表的每个日程项和数据库中的日程信息数据一一对应，向用户展示了日程的标题、时间、完成情况，用户还可以通过点击单个日程项查看或修改日程的详细信息、长按单个日程项实现目标日程的删除；底部的输入框帮助用户添加新的日程，该控件由 ImageButton + EditTextView + ImageButton 的组合方式实现。

图 5.1 系统主界面

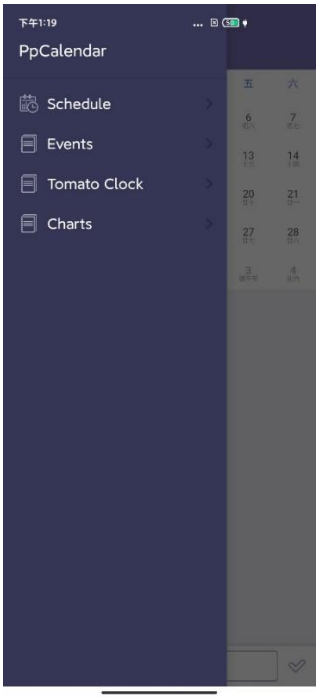

图 5.2 系统菜单栏

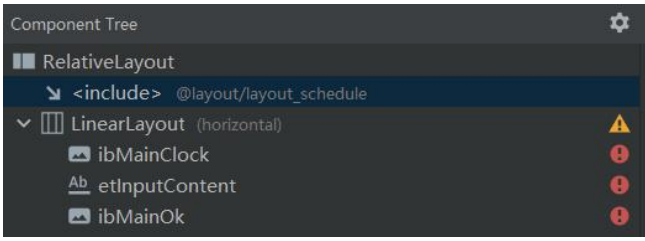

图 5.3 主界面布局树 1

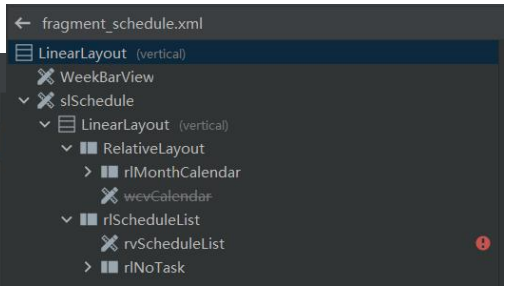

图 5.4 主界面布局树 2

主界面通过 initView()方法用于初始化页面中的各个控件：

```
    private void initView() {
        mDrawer = findViewById(R.id.dlMain);
        btnMenu = findViewById(R.id.btnMenu);
        ……
        btnMenu.setOnClickListener(this);
        tvMainTitle.setOnClickListener(this);
        ……
    =   getResources().getStringArray(R.array.calendar_month);  //获取月份字符串
数组
        llTitleDate.setVisibility(View.VISIBLE);  //初始化标题栏的日期数据
    }
```

日历控件中日期被选择后通过 setCurrentSelectDate()方法进行回调：

```
    private void setCurrentSelectDate(int year, int month, int day) {
        mCurrentSelectYear = year;
        mCurrentSelectMonth = month;
        mCurrentSelectDay = day;
    }
```

通过 gotoScheduleFragment()和 gotoEventSetFragment()方法实现视图的切换：

```
    private void gotoScheduleFragment() {
        FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
        ft.setTransition(FragmentTransaction.TRANSIT_NONE);
        if (mScheduleFragment == null) {
            mScheduleFragment = ScheduleFragment.getInstance();
            ft.add(R.id.flMainContainer, mScheduleFragment);
        }
        if (mEventSetFragment != null) {
            ft.hide(mEventSetFragment);
        }
        ft.show(mScheduleFragment);
        ft.commit();
        llTitleDate.setVisibility(View.VISIBLE);
        tvMainTitle.setVisibility(View.GONE);
        mDrawer.closeDrawer(Gravity.START);
```

5.2 日程管理实现

5.2.1 日程展示

日程管理模块包含日程的增删改查四大功能，具体实现依赖于 ScheduleFragment、EventSetFragment 两大碎片（碎片是 Android 的一个视图组件），这两大碎片的功能相似，区别在于 UI 布局：ScheduleFragment 是以日历视图来展示用户的日程数据，可以通过点击日历上的日期来切换展示不同日期下的日程安排；EventSetFragment 则是日程收集箱，不区分日期，以日程的完成情况为划分依据，将用户所有的日程数据以列表的形式展示。

图 5.5 ScheduleFrag 实现效果

图 5.6 EventSetFrag 实现效果

无论是 ScheduleFragment 还是 EventSetFragment，在展示日程数据时都是依赖于 Android 的 RecyclerView 控件去实现的。

RecyclerView 的初始化代码如下：

```
    private void initScheduleList() {
        rvScheduleList = slSchedule.getSchedulerRecyclerView();
        LinearLayoutManager manager = new LinearLayoutManager(mActivity);
        manager.setOrientation(LinearLayoutManager.VERTICAL);
        rvScheduleList.setLayoutManager(manager);
        DefaultItemAnimator itemAnimator = new DefaultItemAnimator();
        itemAnimator.setSupportsChangeAnimations(false);
        rvScheduleList.setItemAnimator(itemAnimator);
        mScheduleAdapter = new ScheduleAdapter(mActivity, this, scheduleList,
rvScheduleList);
        rvScheduleList.setAdapter(mScheduleAdapter);
    }
```

5.2.2 日程添加

主页底部的输入栏用于帮助用户添加新的日程。用户在中间输入框填写日程的标题后，然后点击右侧添加按钮即可。如果用户在添加日程的同时想设置日程的日期时间，在点击右侧添加按钮之前先通过点击左侧按钮弹出的对话框设置日期和时间即可。

图 5.7 添加日程

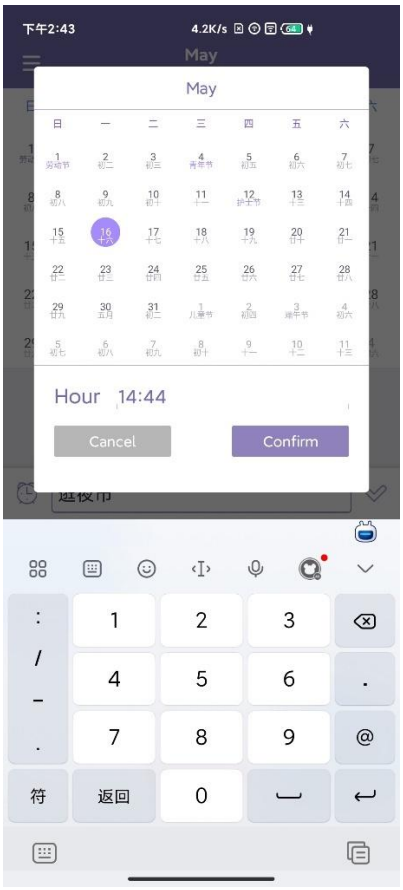

图 5.8 设置日程时间

设置日程的日期和时间时所适使用的对话框并没有使用 Android 提供的原生控件，而是使用 xml 文件自定义实现，其布局树如下图所示：

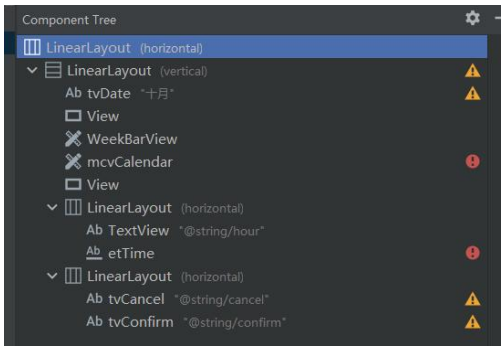

图 5.9 日期选择 Dialog 布局树

添加日程的核心代码如下：

```
    public void insertSchedule() {
        String content = etInputContent.getText().toString();
        if (TextUtils.isEmpty(content)) {
          ToastUtils.showShortToast(mActivity,
R.string.schedule_input_content_is_no_null);
        } else {
          closeSoftInput();
          etInputContent.getText().clear();
          Calendar calendar = Calendar.getInstance();
          calendar.set(mCurrentSelectYear,                mCurrentSelectMonth,
mCurrentSelectDay);
          Schedule schedule = new Schedule();
          schedule.setTitle(content);
          schedule.setDesc("");
          schedule.setYear(mCurrentSelectYear);
          schedule.setMonth(mCurrentSelectMonth);
          schedule.setDay(mCurrentSelectDay);
          schedule.setFinish(false);
          schedule.setTime(mTime);
          mTime = 0;
          if(mScheduleAdapter != null) {
            mScheduleAdapter.insertItem(schedule);
          }
        }
```

5.2.3 日程删除

当系统监听到 ScheduleFrag 或者 EventSetFrag 所展示的日程列表中的某个日程项被长按时，会自动弹出一个删除对话框，帮助用户实现对某个目标日程的删除。

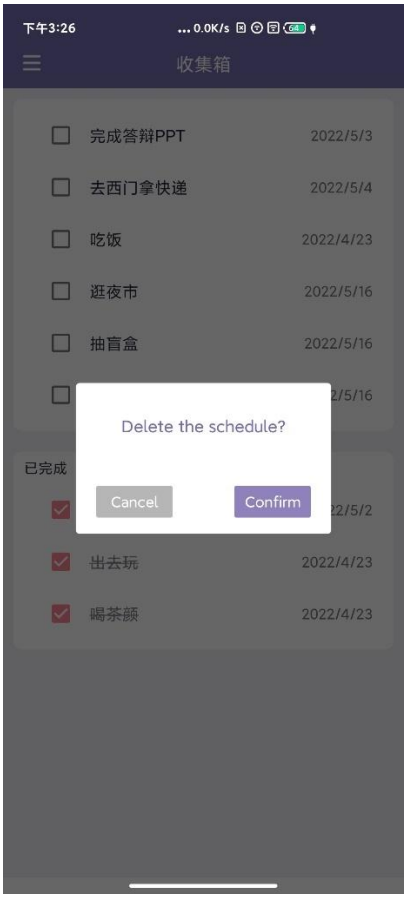

图 5.10 日程的删除

长按 Item 所弹出的删除对话框也没有使用 Android 提供的 Dialog，而是通过 xml 定制实现，其布局树如下：

图 5.11 删除对话框的布局树

删除功能核心代码如下：

```
    public void removeItem(Schedule schedule) {
        final int position = mSchedules.indexOf(schedule);
        if(position != -1) {
          while(!mRv.isComputingLayout() && mRv.getScrollState() ==
RecyclerView.SCROLL_STATE_IDLE) {
            mCheckState.delete(position);
            mSchedules.remove(position);
            ScheduleDao.getInstance().deleteSchedule(schedule.getId());
            notifyItemRemoved(position);
            notifyDataSetChanged();
            mFragment.resetVisibilityOfNoTaskView();
            break;
          }
        }
      }
```

5.2.4 日程修改

如果用户需要对某个日程数据进行修改，可以通过单击日程列表中的目标日程项跳转到日程详情页进行修改操作。日程详情页会展示日程的标题、内容、计划完成时间、完成状态等数据，用户在此页面修改信息后数据会同步更新到数据库中。

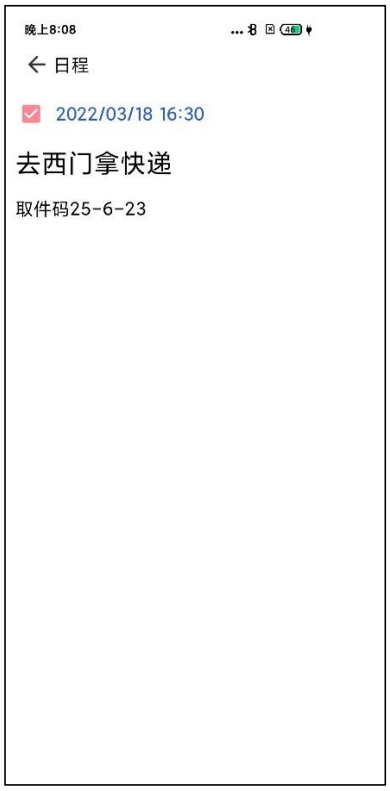

图 5.12 日程详情页

该页面由标题栏和内容栏组成，标题栏左侧包含返回按钮，内容栏中通过 CheckBox 展示日程的完成状态，通过 TextView 展示日程计划完成时间，通过 EditTextView 展示日程的标题和内容，其布局树如下：

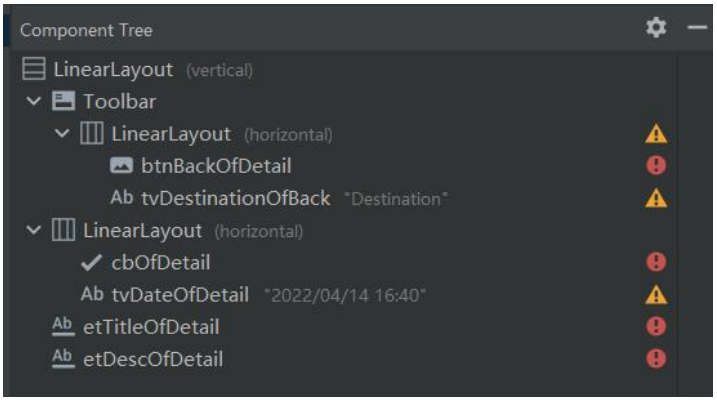

图 5.13 日程详情页布局树

日程修改的核心代码如下：

```
    etTitle.addTextChangedListener(new TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count, int
after) {
            }

            @Override
            public void onTextChanged(CharSequence s, int start, int before, int
count) {
            }

            @Override
            public void afterTextChanged(Editable s) {
               //Log.d(TAG,          "afterTextChanged:          Title="          +
etTitle.getText().toString());
               mSchedule.setTitle(etTitle.getText().toString());
               ScheduleDao.getInstance().updateSchedule(mSchedule);
            }
    });
```

5.3 日程提醒实现

为帮助用户更好地完成日程任务，系统还提供了两种日程提醒方式以及时告知用户待做事项。第一种方式是“到点提醒”：用户为日程设置了计划完成时间后，系统会根据用户所设置的时间在后台创建一个闹钟任务，当到达指定时间后，闹钟任务会调用 Android 系统组件在通知栏发起一个日程通知，提醒用户及时换成任务，用户还可以通过点击通知栏中的通知跳转到日程详情页中查看日程的详细信息；第二种提醒方式是“定时推送”：系统每天八点会将当天用户全部日程任务罗列在通知栏中，提醒用户今天有哪些任务需要完成，通过点击通知栏中的提醒通知，用户可以跳转到系统主页查看当前日期下的日程任务。

图 5.14 到点提醒效果图

图 5.15 定时推送效果图

日程管理模块的核心实现在于 Service 组件的使用，Service 组件是 Android 四大组件之一，它运行在系统的后台，即使应用程序从前台退出，Service 也仍然能正常运行。我们在 Service 中通过 AlarmManager 为每个日程都设置了定时任务，到达指定时间后，AlarmManger 设置的闹钟被执行，紧接着会通知 Service 调用 NotificationManager 发起系统通知提醒用户日程信息。

日程提醒模块的核心代码如下：

```
public class AlarmBinder extends Binder {
        public void addAlarm(long id, long time) {
            Log.d(TAG, "addAlarm: ");
            Intent intent = new Intent();
            intent.setAction(MainActivity.ACTION_ALARM_NOTIFICATION);
            intent.putExtra(ScheduleDetailActivity.SCHEDULE_ID, id);
            PendingIntent pi = PendingIntent.getBroadcast(AlarmService.this, (int)
id, intent, 0);
            am.set(AlarmManager.RTC_WAKEUP, time, pi);
        }

        public void cancelAlarm(long id) {
            Log.d(TAG, "cancelAlarm: ");
            Intent intent = new Intent();
            intent.setAction(MainActivity.ACTION_ALARM_NOTIFICATION);
            intent.putExtra(ScheduleDetailActivity.SCHEDULE_ID, id);
            PendingIntent pi = PendingIntent.getBroadcast(AlarmService.this, (int)
id, intent, 0);
            am.cancel(pi);
        }
    }
```

5.4 番茄钟实现

为帮助用户提高工作效率，系统根据番茄学习法为用户提供了番茄钟功能。在开始某项日程之前，用户可以根据个人情况设置一个倒计时，在倒计时结束之前专注于当前任务，倒计时结束之后休息一会儿再继续任务。系统提供的番茄钟的操作也很简单，通过在屏幕内上下滑动手指可以来实现番茄计时时长的调整，时长设置完毕之后点击开始按钮即可开始计时。番茄计时的数据会以日期为划分依据，同步跟新到数据库中，用户可以在数据反馈页面查看自己的专注情况。

图 5.16 番茄钟实现效果

番茄钟的实现主要通过自定义 View 的形式实现，核心在于 TomatoView 的实现。番茄计时页面的布局树如下：

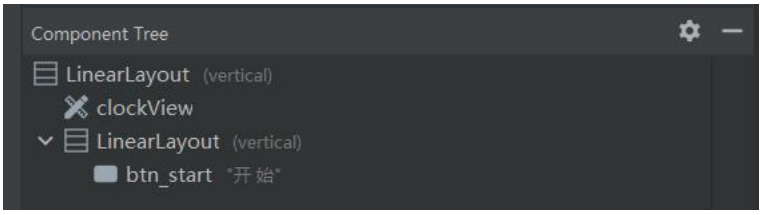

图 5.17 番茄钟页面布局树

番茄钟的主体由外圈的一个随着时间不断消失的黑色圆环以及内圈一个数值不断减小的时间文本组成，无论是视图动画还是滑动调整时长功能都是通过自定义 View 的方式实现的，其核心代码如下：

```
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);
        mRectF.set(centerX - radius, centerY - radius, centerX + radius, centerY + radius);
        mPaint.setAntiAlias(true);  //为 Paint 添加抗锯齿属性
        //黑圆
        canvas.save();
        mPaint.setColor(Color.BLACK);
        mPaint.setStyle(Paint.Style.STROKE);
        mPaint.setStrokeWidth(dpToPixel(5));
        canvas.drawCircle(centerX, centerY, radius, mPaint);
        canvas.restore();
        //灰圆
       ……
        //时间
        canvas.save();
        timePaint.setColor(Color.BLACK);
        timePaint.setStyle(Paint.Style.FILL);
        timePaint.setTextSize(dpToPixel(40));
        canvas.drawText(textTime, centerX - timePaint.measureText(textTime) / 2,
                centerY - (timePaint.ascent() + timePaint.descent()) / 2, timePaint);
        canvas.restore();
    }
```

5.5 数据反馈实现

为帮助用户更好地查看和调整自身状态，系统将收集到的数据从 SQLite 数据库中取出，并借助第三方框架 HelloCharts 将取出的数据以图表的形式在应用程序界面中向用户展示，所展示的数据包括本周的日程完成情况、专注时长数据，以及本月的日程完成情况、专注时长数据。其中系统借助 TextView 将各项具体数据以本文形式展示，将日程完成情况以饼状图的形式展示，将用户专注时长以折线图的形式展示，并通过按钮切换本周、本月的数据。

图 5.18 数据反馈页实现效果

数据反馈页布局树如下：

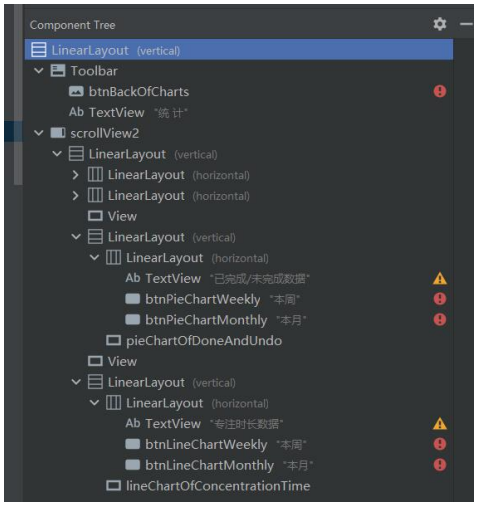

图 5.19 数据反馈页布局树

其核心代码如下：

```
    private void initView() {
        btnBack = findViewById(R.id.btnBackOfCharts);
       ……
        lineChart = findViewById(R.id.lineChartOfConcentrationTime);
        btnShowWeekDataOfLineChart =
findViewById(R.id.btnLineChartWeekly);
        btnShowMonthDataOfLineChart =
findViewById(R.id.btnLineChartMonthly);
        btnBack.setOnClickListener(this);
        btnShowWeekDataOfPieChart.setOnClickListener(this);
        btnShowMonthDataOfPieChart.setOnClickListener(this);
        btnShowWeekDataOfLineChart.setOnClickListener(this);
        btnShowMonthDataOfLineChart.setOnClickListener(this);
    }
```

6 系统测试

系统测试是软件开发过程中必不可少的步骤，通过系统测试能及时发现软件中潜藏的漏洞与隐患，验证系统是否能满足用户需求，保证系统的质量和可靠性。本章是根据实际情况对系统的各模块进行测试与调试。

6.1 测试环境

6.1.1 硬件环境

本次系统测试所用到的硬件环境包括一台笔记本电脑和一台 Android 手机设备，笔记本电脑主要用于运行各种调试工具以及 Android 模拟器，手机设备则用于运行应用程序。各硬件设备的详细配置如下表所示：

表 6-1 系统测试硬件环境详细配置表

设备名称	配置
ASUS 灵耀 S 2 代	CPU：Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 核心处理器数：4 运行内存：8G 操作系统：Microsoft Windows 10 家庭中文版 系统类型：x64-based PC
Redmi K30	CPU：高通骁龙 730G 核心处理器数：8 运行内存：6G 操作系统：Android 10 屏幕尺寸：6.67 英寸 分辨率：2400×1080 像素

6.1.2 软件环境

系统测试所用的软件环境为 JDK 8, Android SDK 10，Android Studio BumbleBee 以及 Android Studio 自带 Debug 工具、模拟器等。

主要测试方法是通过在模拟器和测试手机上运行所开发应用程序，观察各项功能是否正常，是否有不合理或者不符合用户使用习惯的设计出现。为保证版本兼容性，测试手机上所使用的是 Android 10 系统，模拟器上使用的是 Android 5.0 以及 Android 6.0 系统。

6.2 功能测试

表 6-2 功能测试测试用例

序号	所属模块	功能点	测试步骤	预期结果	结果
1	日程管理	添加日程	点击屏幕下方输入框，弹出软件盘；输入日程标题后点击输入框右侧确定按钮。	软键盘收回，当前日期下日程列表中新增一个日程项，标题为刚刚用户输入的内容。	Pass
2		删除日程	长按日程列表中的某一日程项，弹出删除对话框，点击确定按钮。	被选中日程项从当前所展示的日程列表中移除。	Pass
3		修改日程	单击日程列表中某一个日程项，跳转至日程详情页面，修改日程标题、内容或时间，然后返回上一级页面。	上一级页面中该日程项展示的数据为修改后的数据，在再次进入日程详情页后展示的数据为修改后的数据。	Pass
4		查看日程	进入系统主页，点击日历上不同的日期。	日历下方的日程列表数据随着日期的切换而变化。	Pass

续表 6-3

序号	所属模块	功能点	测试步骤	预期结果	结果
5	日程管理	查看日程	点击主页导航栏左侧菜单选项，进入日程收集箱页面。	用户的全部日程以是否完成为划分标准，通过两个列表展示出来。	Pass
6			单击主页或收集箱页面中日程列表里的某一个日程项。	程序跳转至日程详情页，页面正确地展示日程的标题、内容、计划完成时间、完成状态等数据。	Pass
7	日程提醒	到点提醒	在主页新建日程，并设置日程的计划完成时间后单击输入框右侧的确定按钮。	到达计划完成时间时，系统在通知栏发起一个标题为日程标题通知，点击该通知可直接跳转至该日程的日程详情页。	Pass
8		定时推送	在主页通过屏幕底部的输入框创建三个日程，并将日程的计划完成时间设置为第二天。	第二天早上八点系统在通知栏发起一个标题为三个日程标题组合的通知，点击该通知可跳转至系统主页。	Pass
9	番茄计时	番茄钟	通过主页导航栏左侧的菜单进入番茄钟页面，在番茄钟范围内上下滑动调整时间，然后点击开始按钮。	番茄钟开始倒计时，外围黑色圆环在不断消失，内部时长数字在不断减小，直至时长为零时外部黑色圆环完全消失。	Pass
10	数据反馈	图表展示	点击主页导航栏左侧菜单选项，进入统计页面。	该页面中，用户本周及本月的日程完成情况、专注时长等数据以数字或饼状图或折线图的形式被正确地展示。	Pass

6.3 兼容性测试

目前市面上的手机机型丰富、用户所使用的 Android 版本也不尽相同，为保证所研发的系统在不同分辨率上、Android 版本不同的手机设备上都表现一致，能正常、平稳地运行，在系统测试的过程中采用了真机+模拟器的形式，为不同类型设备安装了应用程序进行测试。测试时所用到的设备类型及测试结果如下表所示：

表 6-4 对不同设备测试结果

	设备名称	配置	表现情况
1	Redmin K30	系统版本：Android 10 屏幕尺寸：6.67 英寸 分辨率：2400×1080 像素	正常
2	Nexus 4	系统版本：Android 5.0 屏幕尺寸：4.7 英寸 分辨率：1280×768	正常
3	Piexl 4	系统版本：Android 5.0 屏幕尺寸：5.7 英寸 分辨率：2280×1080	正常
4	Piexl 4	系统版本：Android 6.0 屏幕尺寸：5.7 英寸 分辨率：2280×1080	正常

7 总结

随着社会的进步和互联网技术的高速发展，人们的生活也发生了翻天覆地的变化：纸币被移动支付取代，线下商店被网上商城取代，报纸期刊被公众号营销号取代，这些变化无一不意味着当前社会人们的生活更加方便、舒适。但是在生活幸福感不断提高的同时，人们的生活节奏也在日益加快，每天需要处理的事情也越来越多。在当前快节奏的社会环境下，一款帮助人们提高生活、学习、工作效率的日程管理工具是日常生活中不可或缺的。本文所研究的系统旨在帮助人们合理规划时间、提高做事效率以适应当前日益加快的生活节奏。

本文从所研究系统的需求分析开始，经历了需求分析、方案设计、技术实现、功能测试等步骤，最终实现了一个基于 Android 平台的、功能齐全的日程管理系统。系统包含日程管理、日程提醒、番茄计时、数据反馈等功能模块，基本覆盖了用户日常使用中的全部场景。

当然如果作为一个面向市场的软件该系统还是有些许不足之处，集中表现在两个方面：第一个方面为 UI 设计上的缺陷，因为个人美工技术的不足，系统部分页面的 UI 设计上稍显死板，不够美观；第二个方面则是没有实现数据上云，更换设备后用户数据无法随之一起迁移，即没有实现不同设备间的数据同步。

本课题所设计的 Android 的日程管理系统已成功实现，虽然由于开发周期较短软件本身还存在一些值得优化的地方，但是课题的立题主旨已经顺利实现。在课题的设计和实现过程中，我自身的能力也得到了较大的提升，对软件开发过程的理解也更上一层楼。

参考文献

[1] 中商情报网.2021 年全球智能手机出货量及高端机型市场分析[EB/OL].(2022-3-25)[2022-5-10]. https://baijiahao.baidu.com/s?id=1728202135219072834&wfr=spider&for=pc.

[2] 艾媒网. 手机行业数据分析：2021 年中国网民智能手机操作系统为安卓的占比 89.6%[EB/OL].(2021-8-3)[2022-3-25]. https://www.iimedia.cn/c1061/79273.html.

[3] Statcounter. Mobile Operating System Market Shared in China[EB/OL].(2022-4)[2022-5-10]. https://gs.statcounter.com/os-market-share/mobile/china/#monthly-202004-202104

[4] 安辉. Android App 开发从入门到精通. [M]. 北京：清华大学出版社，2018.12.

[5] 黄日胜. Android 开发基础教程. [M]. 北京：中国水利水电出版社，2018.09.

[6] 刘望舒. ANDROID 进阶之光. [M]. 北京：电子工业出版社，2017.07.

[7] 郭霖. 第一行代码 Android 第 2 版. [M]. 北京：人民邮电出版社，2016.11.

[8] Grant Allen, Mike Owens. SQLite 权威指南（第二版）[M]. 电子工业出版社, 2012.01.

[9] Chuang C Y, Wang Y C, Lin Y B. Digital right management and softeware protection on Android phones[C]. Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st.IEE, 2010.

[10] 赵政, 王彦冰. 基于 MQTT 协议的远程管理智能化 Android 系统设计与实现. [J]. 科学技术创新，2020.

[11] 吕文超, 杨添翔, 邹犇犇, 等. 校园任务与通知发布移动 APP 系统的设计与实现. [J]. 电脑知识与技术，2019.

[12] Bulter M. Android: Changing the mobile landspace[J]. Pervasive Computing, IEEE, 2011.10.

[13] 林晖. 编程全民化，福兮祸兮？. [J]. 中国信息技术教育，2019.

[14] 袁明兰, 王晓鹏, 孔春丽. Java 程序设计. [M]. 北京希望电子出版社，2018.11.

[15] 王韦刚, 张睿. 基于 Android 的参数转换器的设计与实现. [J]. 电子设计工程，2019.

致谢

大学四年时光匆匆，不过弹指一挥间。仿佛昨日才刚刚迈入校园的大门，今朝便得离开这承载了我四年青葱岁月的地方，还未佩妥剑，转眼便江湖。这四年，哭过笑过，吵过闹过。室友第一次见面时的拘谨和小心翼翼、写下第一行代码时的开心和求知若渴、初到公司实习时的好奇和忐忑不安，正是这些毫不起眼的一点一滴交错组合，构成了我大学四年的全部回忆，铸就了我人生迄今为止最得意的时光。

行文至此，思绪万千，百感交集。四年很长，长到曾经我无数次期盼着毕业后的天高任鸟飞；四年很短，短到之前我有多渴望离开，现在就有多渴望着悠悠时光走的再慢一点，让我再多留恋一点。回首过去四年，唏嘘不已。从稚嫩到成熟，从懵懂到明智，从迷茫焦虑到坚定从容，感恩这期间所有的遇见，是你们让我一步一步蜕变为更好的自己。

感恩吾亲，二十三年养育之恩，没齿难忘；感恩吾师，传道受业解惑，如春日夜雨，润物细无声；感恩吾友，陪我走过这段峥嵘岁月，成为点亮我孤独世界的那一抹光明。

今当远离，临表涕临，不知所言。