

**系统能力培养课程**

**课程论文**

学 院： 计算机学院 专 业：计算机科学与技术

小组成员： 张一卓 朱国强 张家齐

选题名称： 云记事本

指导教师： 杨鸣

2022 年 7 月

目录

[一 概述 6](#_Toc107877202)

[二 相关原理与技术 6](#_Toc107877203)

[2.1 MVC架构 6](#_Toc107877204)

[2.2 MVVM架构 8](#_Toc107877205)

[2.3 活动Activity 9](#_Toc107877206)

[2.4 布局Layout 10](#_Toc107877207)

[2.5 Gradle 12](#_Toc107877208)

[2.6 本地持久化 12](#_Toc107877209)

[三 需求分析 14](#_Toc107877210)

[3.1 功能需求 14](#_Toc107877211)

[3.2 性能需求 14](#_Toc107877212)

[四 概要设计 14](#_Toc107877213)

[4.1 开发环境 14](#_Toc107877214)

[4.2 数据结构 15](#_Toc107877215)

[4.3 模块设计 19](#_Toc107877216)

[4.4 接口设计 20](#_Toc107877217)

[4.5 ER图 21](#_Toc107877218)

[4.6 用例图 22](#_Toc107877219)

[4.7 类图 23](#_Toc107877220)

[4.8 流程图 24](#_Toc107877221)

[五 详细设计 24](#_Toc107877222)

[5.1 详细功能设计 24](#_Toc107877223)

[5.2 详细接口设计 27](#_Toc107877224)

[六 系统实现 31](#_Toc107877225)

[6.1 登录与注册 31](#_Toc107877226)

[6.2 用户信息 32](#_Toc107877227)

[6.3 文件管理 33](#_Toc107877228)

[6.4 文件编辑 37](#_Toc107877229)

[6.5 笔记列表 38](#_Toc107877230)

[6.6 设置 39](#_Toc107877231)

[6.7 回收站 39](#_Toc107877232)

[6.8 计划 40](#_Toc107877233)

[6.9 帮助 41](#_Toc107877234)

[6.10 分享 42](#_Toc107877235)

[6.11 外部载入 43](#_Toc107877236)

[6.12 云同步 44](#_Toc107877237)

[6.13 多人协同 45](#_Toc107877238)

[七 测试 45](#_Toc107877239)

[八 总结 46](#_Toc107877240)

[8.1 开发难点 46](#_Toc107877241)

[8.2 协作问题 47](#_Toc107877242)

[8.3 经验教训 47](#_Toc107877243)

[九 参考文献 49](#_Toc107877244)

[十 附录 49](#_Toc107877245)

**摘要**

本文是对云记事本APP Time Taker设计与开发过程的阐述。该APP采用MVVM架构，Android原生态开发，可以运行在Android版本5.0+或使用鸿蒙系统的移动设备上。支持图文混编、分类存取记事、可搜索，既可本地存储又能进行云同步，数据可分享，并具有一定的安全保护功能。相较于市面上的同类软件，Time Taker更加简洁，使用更加便利，更能满足当代年轻大学生的需求。

This paper is an exposition of the design and development process of cloud notepad APP Time Taker. The APP uses MVVM architecture, Android original development, and can run on Android version 5.0 + or mobile devices using HarmonyOS system. Support picture and text mixing, classified access notes, searchable, both local storage and cloud synchronization, data can be shared, and has a certain security protection function. Compared with similar software on the market, TimeTaker is more concise, more convenient to use, and can better meet the needs of contemporary young college students.

**关键词：云记事本、MVVM架构、Android、**

# 概述

在网络高速发展的今天，数据源源不断的从各个媒介传入我们的移动设备，不知不觉我们已经走入一个数据爆炸时代。在海量数据中，如何将自己需要的数据及时保存，如何随身携带所思所想记录生活？在这种需求的支撑下，一大批记事本APP如雨后春笋般冒出来。诸如有道云笔记、印象笔记、我来、OneNote等等，似乎每一款都可以，但又仿佛每一款都差了点什么，很难进行选择。

针对这种情况，Time Taker应运而生，它是一款以更便利地记录为目的的云笔记软件,简洁的界面，丰富的排版，清晰的逻辑，在兼备传统记事本的功能之外，增添了云端备份永久存储、日程管理、Markdown渲染等功能，所有笔记均可导出分享，使用其它办公软件二次编辑，具有市场上大多同类APP所不具备的拓展性。

# 相关原理与技术

## MVC架构

模型-视图-控制器 (MVC) 设计模式为应用程序中的对象分配三个角色之一：模型、视图或控制器。该模式不仅定义了对象在应用程序中扮演的角色，还定义了对象相互通信的方式。这三种类型的对象中的每一种都通过抽象边界与其他对象分开，并通过这些边界与其他类型的对象进行通信。应用程序中特定 MVC 类型的对象集合有时称为层，例如模型层。

MVC 是 Cocoa 应用程序良好设计的核心。采用这种模式的好处很多。这些应用程序中的许多对象往往更具可重用性，并且它们的接口往往被更好地定义。具有 MVC 设计的应用程序也比其他应用程序更容易扩展。此外，许多 Cocoa 技术和架构都基于 MVC，并要求您的自定义对象扮演 MVC 角色之一。

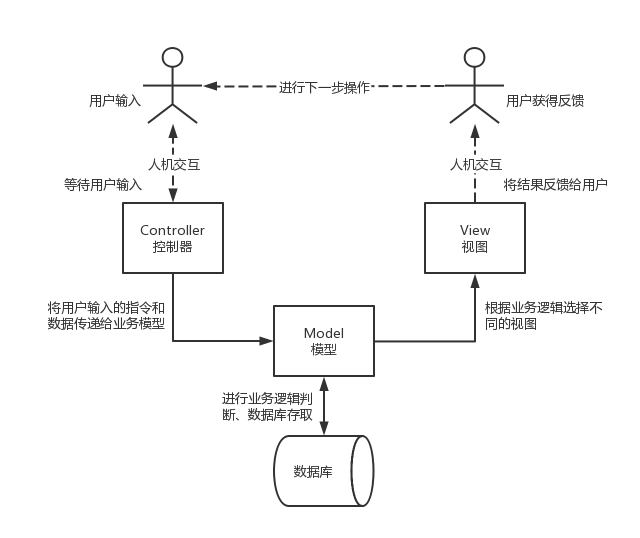


图 二‑1 MVC框架

模型对象（*Model*）

模型对象封装特定于应用程序的数据，并定义操作和处理该数据的逻辑和计算。例如，模型对象可能代表游戏中的角色或地址簿中的联系人。一个模型对象可以与其他模型对象具有一对一和多对多的关系，因此有时应用程序的模型层实际上是一个或多个对象图。

在数据加载到应用程序后，作为应用程序持久状态一部分的大部分数据（无论持久状态存储在文件中还是数据库中）都应该驻留在模型对象中。因为模型对象代表与特定问题域相关的知识和专业知识，所以它们可以在类似的问题域中重复使用。

查看对象（*View*）

视图对象是应用程序中用户可以看到的对象。视图对象知道如何绘制自己并且可以响应用户操作。视图对象的一个主要目的是显示来自应用程序模型对象的数据并允许编辑该数据。尽管如此，视图对象通常与 MVC 应用程序中的模型对象分离。

因为通常会重用和重新配置它们，所以视图对象提供了应用程序之间的一致性。UIKit 和 AppKit 框架都提供了视图类的集合，Interface Builder 在其库中提供了数十个视图对象。

控制器对象（*Contoller*）

控制器对象充当一个或多个应用程序的视图对象与其一个或多个模型对象之间的中介。因此，控制器对象是一个管道，视图对象通过它了解模型对象的变化，反之亦然。控制器对象还可以为应用程序执行设置和协调任务，并管理其他对象的生命周期。

## MVVM架构

MVC固然很好，但是随着App应用功能的强大Controller的负担越来越大因此在MVC的基础上繁衍出了MVVM框架。

由于Controller主要用来处理各种逻辑和数据转化，复杂业务逻辑界面的Controller非常庞大，维护困难，所以有人想到把Controller的数据和逻辑处理部分从中抽离出来，用一个专门的对象去管理，这个对象就是ViewModel，是Model和Controller之间的一座桥梁。当人们去尝试这种方式时，发现Controller中的代码变得非常少，变得易于测试和维护，只需要Controller和ViewModel做数据绑定即可，这也就催生了MVVM的热潮。

在MVVM框架中VM层（视图模型层）通过接口从后台M层（Model层）请求数据，VM层继而和V（View层）实现数据的双向绑定。

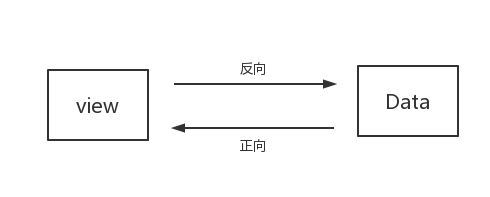


图 二‑2 双向绑定

双向是指ViewModel中的Data部分和View之间的双向关系。正向指数据驱动页面，反向是指页面更新数据，绑定是指自动化处理，Data改变了View随之改变，反之也是。

因此相较于MVC框架，在MVVM框架中1、Controller清晰简洁，ViewModel分离了大部分Controller代码，更加清晰和容易维护。2、方便测试，开发中大部分Bug来自于逻辑处理，由于ViewModel分离了许多逻辑，可以对ViewModel构造单元测试。3、开发解耦，一个人负责逻辑实现、另一个人负责UI实现，敏捷开发时，会经常发不是等后端做好了接口我们再去开发，不过在没有接口的情况下通常我们可以把Controller和View完成。

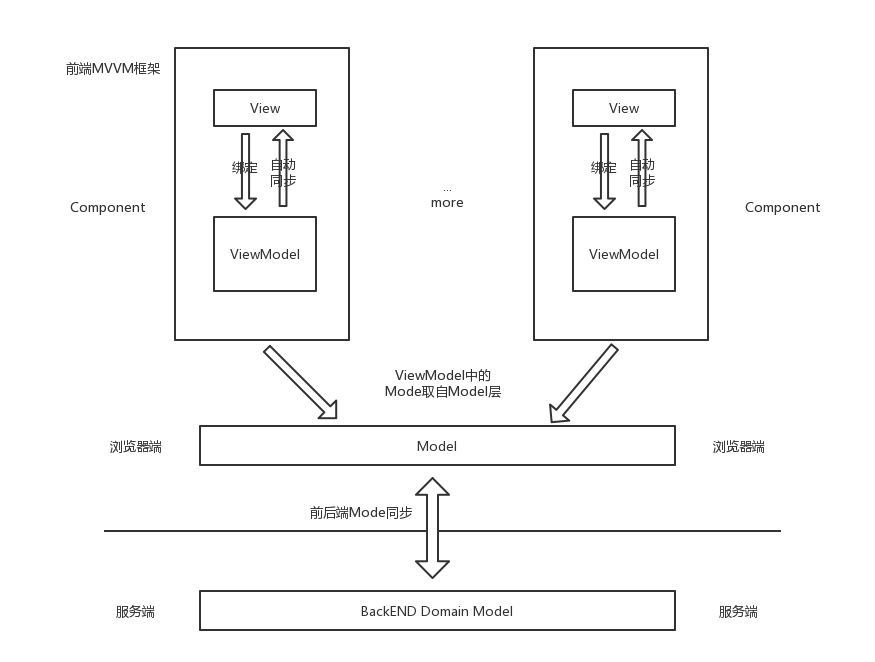


图 ‑ MVVM框架

## 活动Activity

Activity是Android应用的重要组件，多数的编程语言中，程序是通过main()函数来启动的，但是在Android项目中我们找不到main()函数，这是因为Android App的启动方式为：通过Android系统来调用与其生命周期特定阶段相对应的特定回调方法来启动Activity实例中的代码。例如，当一个App被点击启动时，Android系统会创建你写的某个Acticity的实例，并调用其生命周期中的应得onCreate()和onStart()等方法。而创建一个活动需要包括两个步骤1、编写相应的Activity类（Java文件）。2、在AndroidManifest.xml文件中注册。

## 布局Layout

Android共有六大基本布局，分别是线性布局（LinearLayout）、相对布局（RelativeLayout）、层布局（FrameLayout）、网格布局（GridLayout）、表格布局（TableLayout）、绝对布局（AbsoluteLayout）。因在开发过程中仅用到前四个布局，所以只对前四个布局进行介绍。

### 线性布局（*LinearLayout*）

线性布局在开发中使用最多，具有垂直方向与水平方向的布局方式，通过设置属性“android:orientation”控制方向，属性值垂直（vertical）和水平(horizontal)，默认水平方向。

android:gravity：内部控件对齐方式，常用属性值有center、center\_vertical、center\_horizontal、top、bottom、left、right等。这个属性在布局组件RelativeLayout、TableLayout中也有使用，FrameLayout、AbsoluteLayout则没有这个属性。

center：居中显示，这里并不是表示显示在LinearLayout的中心，当LinearLayout线性方向为垂直方向时，center表示水平居中，但是并不能垂直居中，此时等同于center\_horizontal的作用；同样当线性方向为水平方向时，center表示垂直居中，等同于center\_vertical。

top、bottom、left、right顾名思义为内部控件居顶、低、左、右布局。

这里要与android:layout\_gravity区分开，layout\_gravity是用来设置自身相对于父元素的布局。

android:layout\_weight：权重，用来分配当前控件在剩余空间的大小。使用权重一般要把分配该权重方向的长度设置为零，比如在水平方向分配权重，就把width设置为零。

### 相对布局（*RelativeLayout*）

相对布局可以让子控件相对于兄弟控件或父控件进行布局，可以设置子控件相对于兄弟控件或父控件进行上下左右对齐。

RelativeLayout能替换一些嵌套视图，当我们用LinearLayout来实现一个简单的布局但又使用了过多的嵌套时，就可以考虑使用RelativeLayout重新布局。相对布局就是一定要加Id才能管理。

### 层布局（*FrameLayout*）

层布局，从屏幕左上角按照层次堆叠方式布局，后面的控件覆盖前面的控件。该布局在开发中设计地图经常用到，因为是按层次方式布局，我们需要实现层面显示的样式时就可以采用这种布局方式，比如我们要实现一个类似百度地图的布局，我们移动的标志是在一个图层的上面。

### 网格布局（*GridLayout*）

作为android 4.0 后新增的一个布局,与前面介绍过的TableLayout(表格布局)其实有点大同小异，不过新增了一些东西。

1、跟LinearLayout(线性布局)一样,他可以设置容器中组件的对齐方式

2、容器中的组件可以跨多行也可以跨多列(相比TableLayout直接放组件,占一行相比较)

常用属性如下:

排列对齐:

1、设置组件的排列方式: android:orientation="" vertical(竖直,默认)或者horizontal(水平)

2、设置组件的对齐方式: android:layout\_gravity="" center,left,right,buttom

设置布局为几行几列:

1、设置有多少行: android:rowCount="4" //设置网格布局有4行

2、设置有多少列: android:columnCount="4" //设置网格布局有4列

设置某个组件位于几行几列

1、组件在第几行: android:layout\_row = "1" //设置组件位于第二行

2、组件在第几列: android:layout\_column = "2" //设置该组件位于第三列

设置某个组件横跨几行几列（合并）:

横跨几行: android:layout\_rowSpan = "2" //纵向横跨2行

横跨几列: android:layout\_columnSpan = "3" //横向横跨2列

android:layout\_gravity="fill"填充

## Gradle

Gradle 是一种以 Groovy 语言为基础的自动化构建工具，一般通过修改 build.gradle 脚本来完成对项目构建的一些设置，例如依赖管理等等。大多数情况下，只需要稍微修改下 gradle 文件即可完成自己的需求。

自动化构建工具本质上来说也是一种程序，跟自己写的代码一样，开始编译时就启动这个程序，然后读取在 gradle 文件中配置的参数来实例化各个类，然后按照顺序依次执行对应的任务即可完成整个构建任务。

所以 build.gradle 文件，或者其他后缀为 gradle 的文件其实就是个配置文件，就好像 xml 一样，我们在 gradle 文件中修改各种配置参数，Gradle 通过这些参数来实例化 Project 等等就像构造器一样。

当我们新建一个项目后，Gradle 默认会生成一些编译脚本文件，主要有：setting.gradle、build.gradle 以及子项目中的 build.gradle 等等，还会在当前目录下生成一个 gradle 文件夹，下面分别介绍一些这些文件的作用：

setting.gradle 用来告诉 gradle 这个项目包含了哪些子项目。

build.gradle 是默认的构建脚本，当我们在执行 gradle 命令时，会首先到当前目录下寻找该文件，然后通过该文件的配置实例化一个 Project 对象。

自动生成的 gradle 文件夹是 Gradle 包装器，其中包含一个 jar 文件和一个 配置文件，使用这个包装器可以让 Gradle 运行在一个特定的版本上，目的是创造一个独立于系统、系统配置和 Gradle 版本的可靠和可重复构建。

Gradle 中有两个重要的概念，分别是 Project 和 Task，Project 表示一个正在构建的项目，而 Task 表示某一个具体的任务

## 本地持久化

持久化技术就是将数据保存到存储设备中，持久化技术提供了一种机制，可以让数据在瞬时状态和持久状态间进行转换，Android中提供了三种方式用于简单地实现数据持久化功能：文件存储、SharedPreferences存储以及数据库存储。

### 文件存储

文件存储比较适合存储一些简单的文本数据或者二进制数据，如果想用文件存储的方式保存一套较为复杂的结构化数据，就需要定义一套格式规范，方便之后从文件中重新解析出来。

### *SharedPreferences*存储

SharedPreferences是使用键值对的方式来存储数据的，也就是说，当存储一条数据的时候，需要给这条数据提供对应的键，之后就可以通过键将值取出来。SharedPreferences还支持不同类型的数据存储，如存储的时候是整型，则取出时也是整型。它又分为两种Context类中的getSharedPreferences()方法、Activity类中的getPreferences()方法。

Context类中的getSharedPreferences()方法。此方法接收两个参数，第一个指定SharedPreferences文件名称，如果指定文件不存在则会创建一个。默认存储在data/data//shared\_prefs目录下。第二个参数指定操作模式，目前只有默认的MODE\_PRIVATE可选，，它与直接传0的效果是一样的，表示只有当前的应用程序才能对这个文件读写。其他的如MODE\_WORLD\_WRITEABLE、MODE\_WORLD\_READABLE、MODE\_MULTI\_PROCESS均已被废弃。

Activity类中的getPreferences()方法。这个方法和上面的很类似，但只接收一个文件操作模式，因为getPreferences()方法会自动将当前ACtivity的类名作为文件名。得到一个SharedPreferences对象后，就可以开始向SharedPreferences文件中存储数据了。主要分为3步：1、调用SharedPreferences对象的edit()方法获取一个SharedPreferences.Editor对象。2、然后向这个Editor对象中添加数据，格式是putBoolean()、putString()等等。3、最后调用Editor的apply()方法将数据提交，从而完成数据存储操作。

而读取数据的方法很简单，首先要构造一个SharedPreferences对象，然后调用诸如getInt()、getString()方法,这些方法接收两个参数，第一个是键，第二个是默认值，如果没找到这个键将会返回这个默认值。

### *SQLite*数据库存储

Android系统内置了SQLite这款轻量的关系型数据库，占用资源少，通常只需要几百KB的内存就够了。Android专门提供了一个SQLiteOpenHelper的帮助类用于管理数据库。SQLiteOpenHelper是一个抽象类，我们需要去继承它，且必须重写两个抽象方法：onCreate()和onUpgrate()。在这两个方法中可以实现创建和升级数据库的逻辑。

SQLiteOpenHelper中有两个重要的实例方法：getReadableDatabase()和getWritableDatabase()。这两个方法都可以创建或打开一个现有的数据库，并返回一个可对数据库进行读写操作的对象。不同的是，当数据库不可写入的时候，getReadableDatabase()方法返回的对象将以只读的方式打开数据库，而getWritableDatabase()方法将会出现异常。

SQLiteOpenHelper类中有两个构造函数，这里一般使用参数少一点的那个即可：这个构造函数接收四个参数：第一个是context，第二个是数据库的名称，第三个参数允许我们在查询数据的时候返回一个自定义的Cursor，一般传入null即可，第四个参数表示当前数据库的版本号，可用于对数据库进行升级操作。构建出SQLiteOpenHelper的实例后，再调用getReadableDatabase()或getWritableDatabase()就可以创建数据库了，此时重写的onCreate()方法也能得到执行，所以通常在此处处理一些创建表的逻辑。数据库文件会放在/data/data/<package\_name>/databases/目录下。

# 需求分析

## 功能需求

* 支持类Markdown语法,记事内容可以图文混排，可编辑可修改，打印。
* 块,页面,数据库的页面结构,可分类存取记事，可搜索，用本地文件存储/模拟数据库。
* 网络文件存储/网络数据库存储（云存储）。
* 笔记内容可导出,用户之间数据可分享，具有安全保护功能,支持多人协同编辑。
* 适配平板与手机,日程管理作为特殊块整合入笔记。

## 性能需求

* 时间特性要求：一般操作的响应时间在1-2秒内。
* 适应性：满足运行环境，在市面上100%的安卓机上都能运行。
* 灵活性：较灵活的接口，可适应突然变化。

# 概要设计

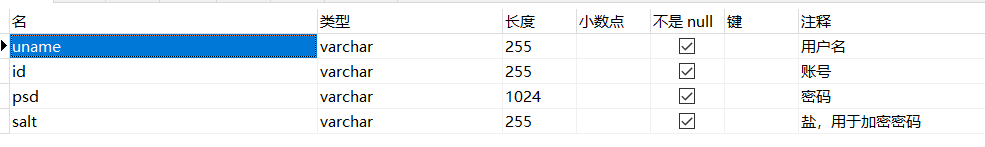
## 开发环境

* Andriod 11
* Gradle 7.2
* SDK 32
* JDK 1.8
* Kotlin 1.7
* Navicat 16
* IDEA 2021.2.3
* AndroidStudio 2021.1.1
* 华为云服务器
* 适配安卓版本5.0+

## 数据结构

### 数据库

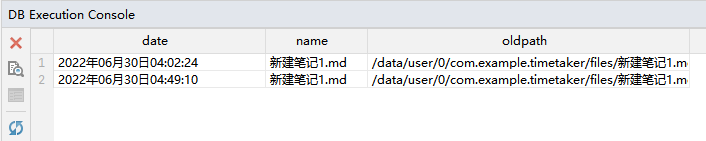
用户资料表的设计：



用户权限表的设计：



回收站recycle.db



云同步Nodes.db

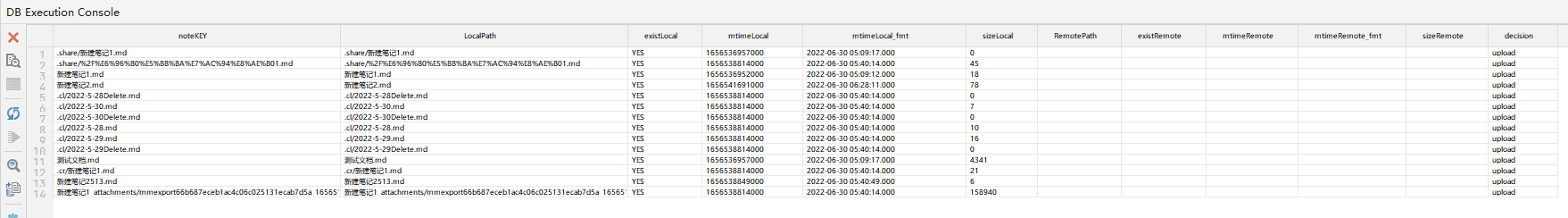


图 ‑ 数据库设计

### 类文件

|  |  |  |
| --- | --- | --- |
| 功能 | 类 | 作用 |
| 登录 | LoginActivity | 登录 |
| UserBean | 用户的类 |
| UserBeanAdapter | 接收数据，更新到  UserBean中 |
| 注册 | RegisterActivity | 进行账号注册 |
| 加密 | PBKDF2 | 加密密码 |
| 用户信息 | UserInformation | 用户信息操作 |
| 修改密码 | ModifyPsdActivity | 修改密码 |
| 云同步 | RetrofitHelper | 单例Retrofit对象 |
| nodesSqliteControl | 操作本地数据库 |
| Nodes | 同步文件类 |
| File\_Syc | Wfj文件上传类 |
| QuotesApi | 云同步相关接口 |
| 文件管理 | CustomComparator | 文件夹/文件排序 |
| CustomFileFilter | 获取文件夹/文件名 |
| FileOpreationUitl | 文件夹/文件  操作工具类 |
| FileSort | 文件夹/文件排序  的具体实现 |
| FragmentAdapter | 文件夹/文件显示  适配器 |
| FragmentFolder | 文件夹/文件管理实现 |
| 计划（日程管理） | CalendarAdapter | 未完成计划适配器 |
| CalendarAdapter2 | 已完成计划适配器 |
| CalendarFragment | 计划模块具体实现 |
| 帮助 | HelpFragment | 帮助模块具体实现 |
| HelpViewModel | 帮助的业务逻辑 |
| 设置ｄｓｆ | SettingsFragment | 设置模块具体实现 |
| ｄｄｄ编辑器 | ｄｓｆｓｄｆｓｄｆEditFragment | 编辑器页面 |
| EditorViewModel | 编辑器数据与业务逻辑 |
| WYSIWYG | 继承自Webview的编辑器控件类 |
| Utils | 工具类 |
| 通用类 | CommonViewModel | 各fragment的共有业务逻辑 |
| 主活动 | Main2Activity | 主活动 |
| MainActiViewModel | 主活动的业务逻辑 |

### 自导入依赖

* MDTool-1.2.5.beta.jar
* mysql-connector-java-5.1.46-bin.jar
* com.kotcrab.remark:remark:1.0.0

## 模块设计

### 名称

* 登录
* 注册
* 用户信息
* 修改密码
* 文件管理
* 文件编辑
* 设置
* 回收站
* 计划
* 帮助
* 分享
* 外部载入
* 云同步
* 多人协同

### 模块关系

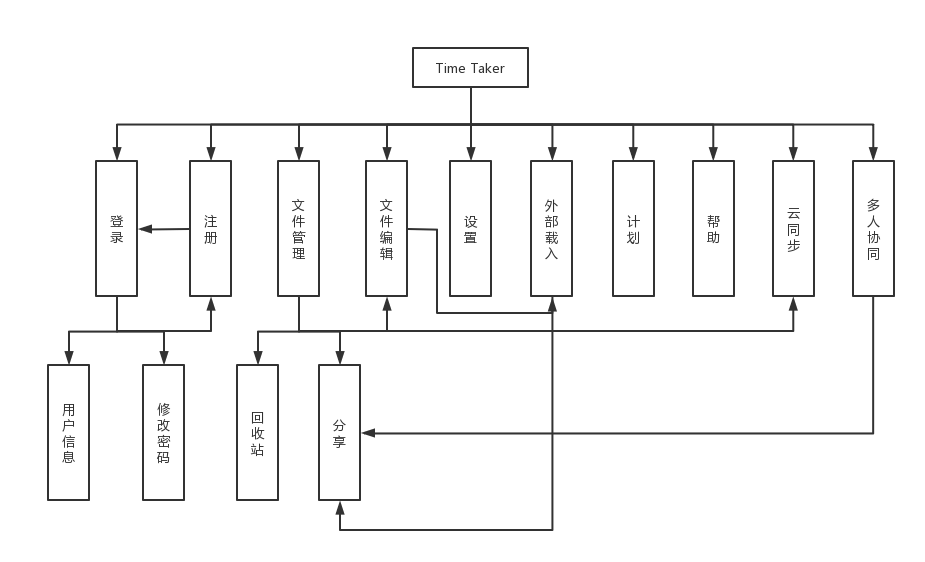


图 ‑模块调用关系图

## 接口设计

详细接口设计见后文接口设计篇

|  |  |  |
| --- | --- | --- |
| 功能 | 接口地址 | 请求方式 |
| 登录 | /login | POST |
| 注册 | / register | POST |
| 获取云端所有笔记信息 | /get\_all\_notes | GET |
| 删除文件 | / delete\_note | GET |
| 上传文件 | /upload/multipart | POST |
| 下载文件 | /download | GET |

## ER图

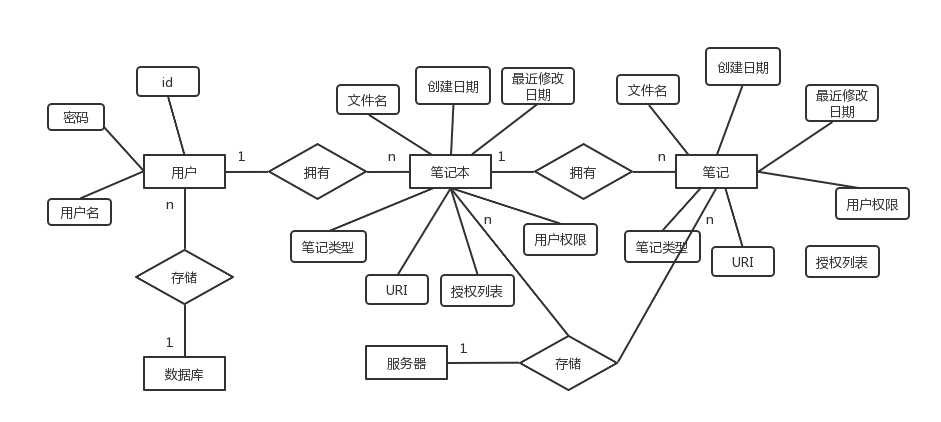
****

图 ‑ 总体ER图

## 用例图

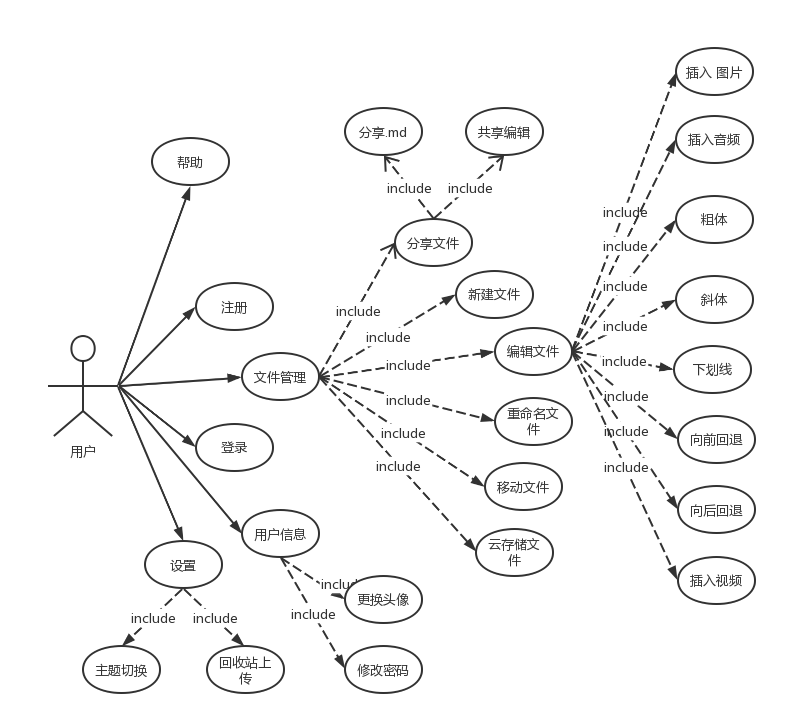
****

图 ‑总体用例图

## 类图

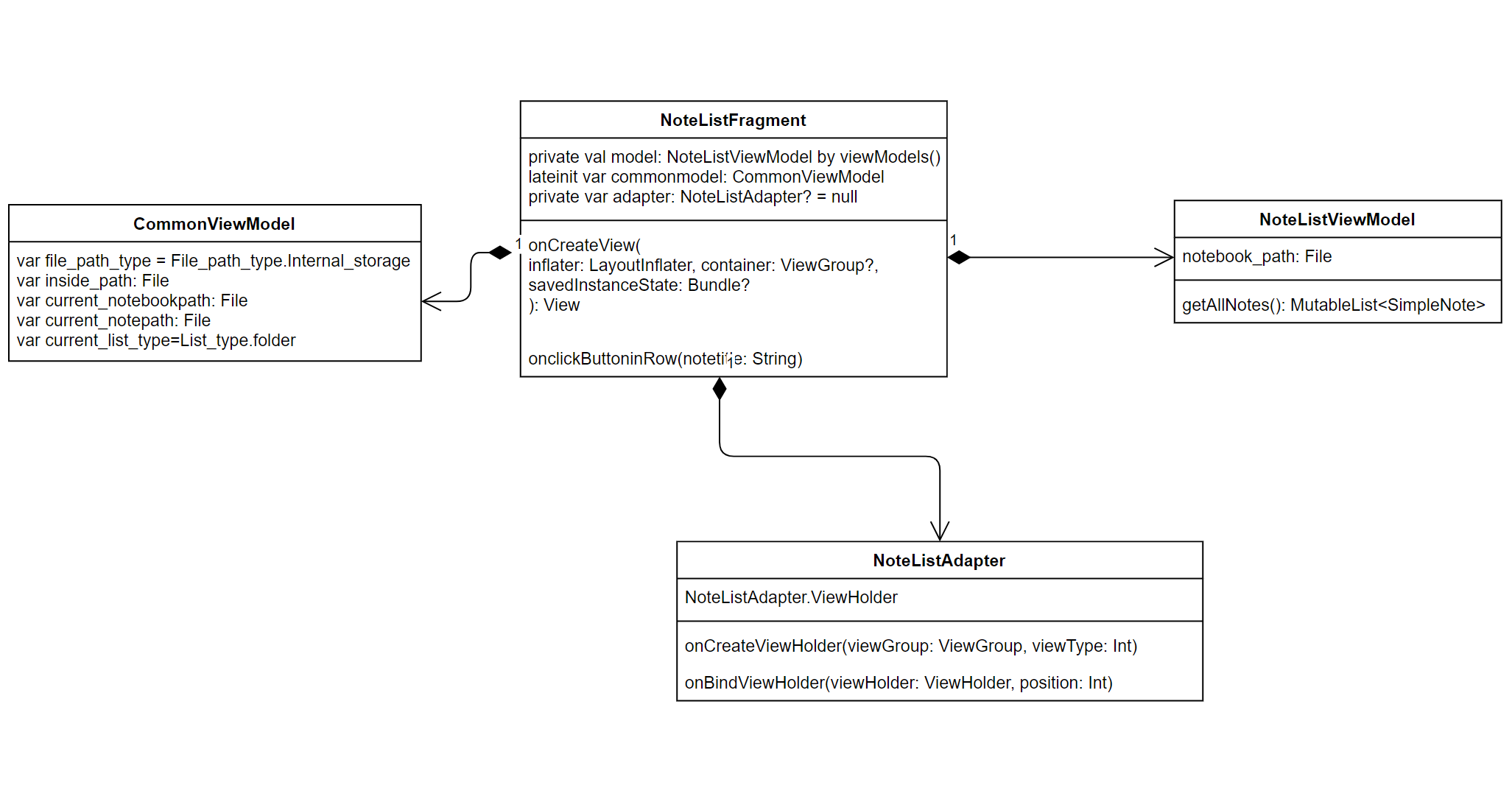


图 ‑ 笔记列表类图

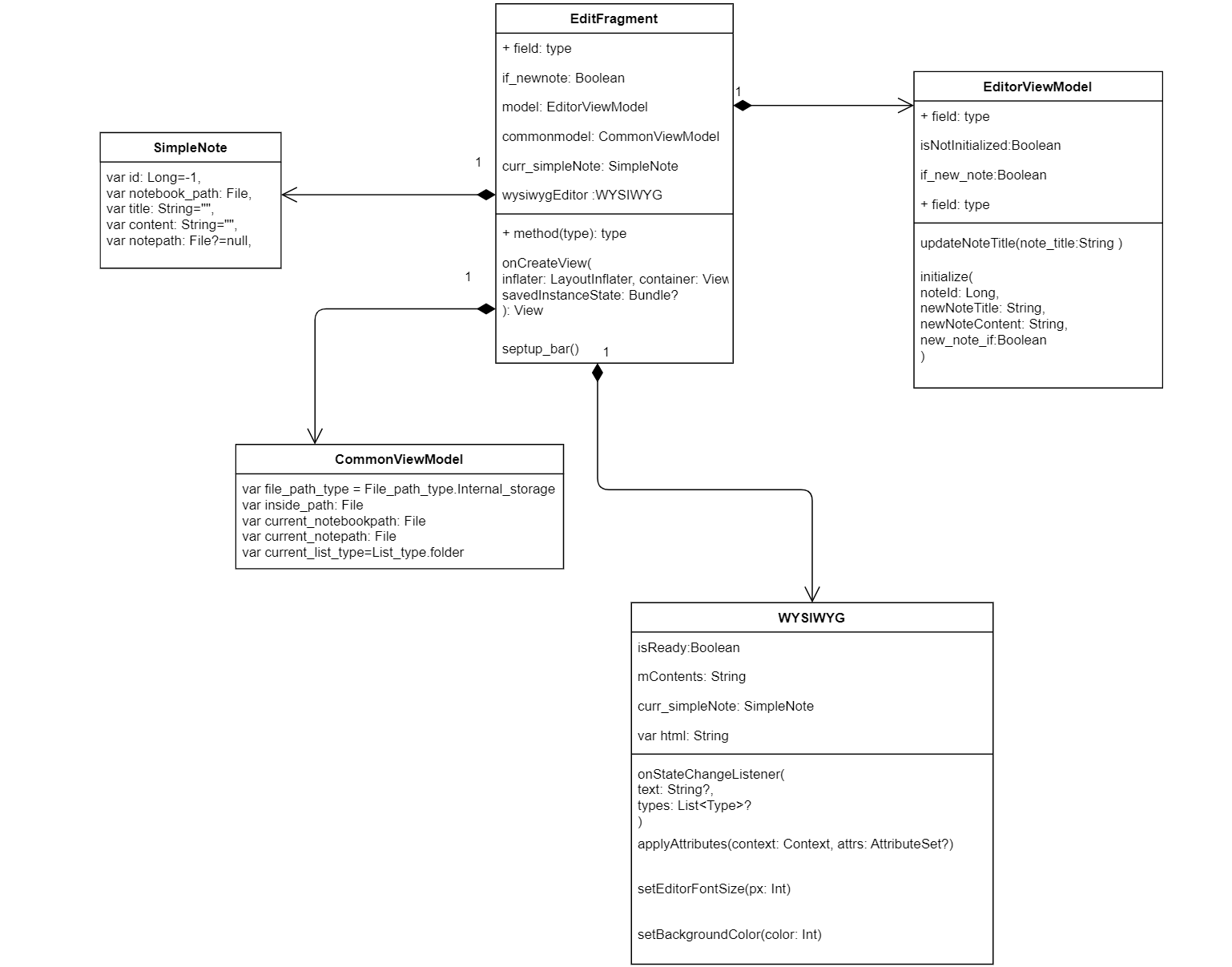


图 ‑ 编辑器类图

## 流程图

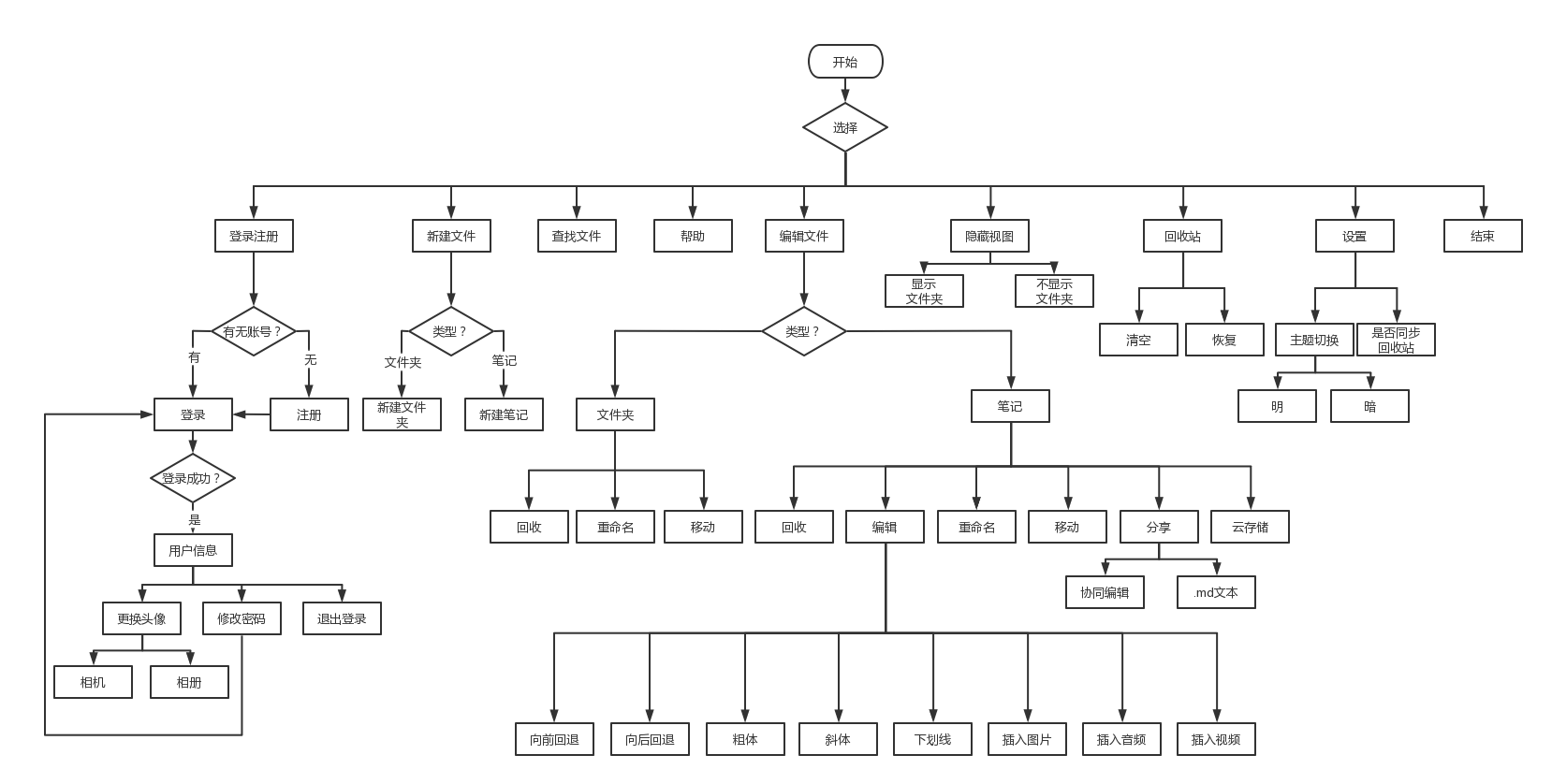
****

图 ‑流程图

# 详细设计

## 详细功能设计

### 登录与注册

登录与注册主要包括登录账号以及注册账号两个功能，在登录界面，用户输入账号和密码进行登录，在点击登录按钮后，判断账号和密码是否为空，为空则进行提示，不为空则判断账号密码是否正确，账号密码都正确则登陆成功，进行提斯，跳转到应用的主界面。在注册页面，用户输入账号和密码并且确认密码之后进行注册，同样判断账号和密码是否为空以及两次输入的密码是否一致，全都没问题则提示注册成功，自动跳转到登录界面。在登录与注册界面，输入账号和密码时点击清除按钮来清除输入的信息，在输入密码时，点击眼睛图标来显示/隐藏密码。

### 用户信息

点击侧边栏头像，根据登录状态来决定是进入到登录界面还是用户资料界面，在用户资料界面显示用户头像、ID、昵称、修改密码、退出登录等信息。用户点击用户资料界面的头像可修改头像，点击修改密码则进入修改密码页面，在输入原密码与新密码后点击确定即可修改密码。密码修改后，用户需要重新登陆。用户点击退出退出登录按钮，则退出登录。

### 文件管理

文件管理主要包括的文件夹的建立，文件夹的重命名，文件夹的回收，文件的重命名、回收、移动。通过文件管理，用户可以按照自己的需要，建立不同的文件树结构，方便文件的整理和归类。用户初次安装APP会在Android设备内部存储区建立根文件夹TimeTaker,此后所有的文件管理均在该文件夹下进行。

### 笔记编辑

笔记编辑模块主要包括Markdown渲染，用户可以对内容进行加粗，斜体，高亮，一级标题，二级标题，添加图片，添加音频，添加视频等操作。同时用户可以对笔记文件进行包括：导出，分享，搜索，协同编辑等操作。

### 笔记列表

笔记列表模块的功能是显示当前选中笔记本下的所有笔记及其笔记预览，相比文件夹结构用户可以更加方便的在不进入笔记内部的情况下了解笔记内容。

### 设置

在设置部分主要实现两大设置，第一个设置是主题的切换，APP中设置明暗两个主题模板，用户可自行选择，并将选择持久化到本地。另一个设置是选择是否在清空回收站时将云端回收站的数据同步删除。

### 回收站

回收站只能进行两个操作，第一个是文件彻底删除，第二个是文件恢复，将选中的文件或者是文件夹恢复到它原来的位置，如果原来的位置发生修改，则新建路径下所有文件夹。

### 计划

计划部分主要是需要实现一个日程管理。这个日程管理并不是对用户笔记记录这一日程进行管理。而是类似便签的情况，用户可以选择日历上的某一天，然后在这一天新建日程计划，点击未完成计划，该计划会被标记为已完成，点击已完成计划，该计划会重新回到未完成列表中。

### 帮助

在帮助部分，设计的是当用户点击，会进入到一个技术博客，用户在该页面点击搜索就可以查找自己遇到的问题，或者进行留言，等待维护团队解决。

### 分享

笔记分享共有三种实现方案，包括调用第三方SDK，或者是平台专属SDK，但这里我们设计时选择调用系统的分享功能，这样可以将笔记分享至设备中所有支持分享功能的APP中，包括微信、QQ、微博等。

### 外部载入

考虑到市面上同类APP均不支持第三方APP二次编辑，我们设置了文件外部载入功能。这里载入的文件要求是Markdown格式，APP载入该文件会自动将文件添加至指定文件夹，点击即可编辑。编辑完成使用分享功能分享出去，该文件还能由其它APP打开再次编辑。

### 云同步

作为一个云记事本，云同步功能也是系统的核心功能之一。当用户登录后，点击“同步所有笔记按钮”后可以将本地笔记与账号下的云端笔记进行比对，依据同步计划对不同笔记进行上传、下载、移动、删除等策略。从而实现在不同手机终端上通过登录相同账号，查看相同笔记与日程的功能。

### 多人协同

本APP实现了较为基础的多人协同功能，用户在登录后，可以选择笔记设置笔记权限（公开或只对某些用户可见）进行生成笔记链接，另一个用户在登录后可以通过他人分享的笔记链接进行笔记的协同编辑。

### 搜索

因为APP是基于文件系统的，如果文件夹内再建文件夹，形成嵌套，就不容易找出文件所在地，所以需要查找功能，只需要输入关键词，就能将所有含有关键词的文件罗列出来，点击即可进入编辑。

## 详细接口设计

### 获取云端所有笔记信息

#### 接口说明

为了编排同步计划，从云端获取所有笔记信息

#### 接口地址

/get\_all\_notes

#### 请求方式

GET

#### 请求参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 是否必传 | 传参示例 |
| Authorization | String | 用户验证信息 | 是 | 4a1d123a31349 |

#### 响应参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 参数示例 |
| success | Bolean | 响应编码 | true |
| message | String | 响应消息 | “ok” |
| notes\_length | Int | 云端笔记数 | 2 |
| note | Object | 响应实体 | [{ RemotePath: "./test.md",  existRemote: "YES",   mtimeRemote: "1243432424",   sizeRemote: "1124",   }] |
| RemotePath | String | 笔记远程路径 | "./test.md" |
| existRemote | String | 笔记云端是否存在 | "YES" |
| mtimeRemote | String | 笔记云端最后修改时间 | "1243432424" |
| sizeRemote | String | 笔记云端大小 | "1124" |

#### 返回示例

get\_all\_notes   
{   
 success: true,   
 notes\_length: 2,   
 notes: [   
 {   
 RemotePath: '/新建笔记2513.md',   
 existRemote: 'YES',   
 mtimeRemote: 1656941039813.521,   
 sizeRemote: 6   
 },   
 {   
 RemotePath: '/测试文档.md',   
 existRemote: 'YES',   
 mtimeRemote: 1656941039791.5208,   
 sizeRemote: 4341   
 }   
 ]   
}

### 删除文件

#### 接口说明

删除云端文件

#### 接口地址

/delete\_note

#### 请求方式

GET

#### 请求参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 是否必传 | 传参示例 |
| Authorization | String | 用户验证信息 | 是 | 4a1d123a31349 |
| note\_path | String | 待删除笔记路径 | 是 | "新建笔记2.md" |

#### 响应参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 参数示例 |
| success | Bolean | 响应编码 | true |
| message | String | 响应消息 | “ok” |

#### 返回示例

如果删除成功

{   
 success: false,   
 message: "成功"   
}

如果删除失败

{   
 success: false,   
 message: "文件不存在"   
}

### 下载文件

#### 接口说明

下载云端文件

#### 接口地址

/download

#### 请求方式

GET

#### 请求参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 是否必传 | 传参示例 |
| Authorization | String | 用户验证信息 | 是 | 4a1d123a31349 |
| note\_path | String | 待下载笔记路径 | 是 | "新建笔记2.md" |

#### 响应参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 参数示例 |
| success | Bolean | 响应编码 | true |
| message | String | 响应消息 | “ok” |

#### 返回示例

如果下载成功

{  
 success: false,  
 message: "成功"  
}

如果下载失败

{  
 success: false,  
 message: "文件不存在"  
}

### 上传文件

#### 接口说明

上传文件到云端

#### 接口地址

/upload/multipart

#### 请求方式

POST

#### 请求参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 是否必传 | 传参示例 |
| Authorization | String | 用户验证信息 | 是 | 4a1d123a31349 |
| target\_path | String | 待上传笔记路径 | 是 | "./测试/新建笔记2.md" |

#### 响应参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 | 参数示例 |
| success | Bolean | 响应编码 | true |
| message | String | 响应消息 | “ok” |

#### 返回示例

如果上传成功

{  
 success: false,  
 message: "成功"  
}

如果上传失败

{   
 success: false,   
 message: "上传失败"   
}

# 系统实现

## 登录与注册

输入账号与密码的文本框采用的是EditText，用户输入账号与密码后，点击登陆按钮时，先开一个子线程，然后获取控件中输入的数据，然后判断账号与密码是否有空的，如果有空的则进行提示，不为空则将账号密码与数据库中的数据进行对比，账号密码都正确则登录成功，提示欢迎，并持久化登录状态。

如果用户点击登录界面的注册按钮，则进入到注册页面，输入账号密码后，先获取到EditText控件中的内容，然后判断是否为空，接下来与登录相同。在登录与注册的EditText控件旁边显示“×”，点击即可清除当前的输入内容，通过添加单击响应事件，当点击时将对应内容置为空即可。同时在输入密码的一栏，为了安全则需要添加显示/隐藏密码按钮，同时为了确保用户数据的安全性，在数据库中存储密文，我们通过使用PBKDF2加密算法将用户的密码加密为密文存放在数据库当中。具体界面如下：

图 ‑ 登录与注册

## 用户信息

点击侧边栏的头像后根据当前是否登录来选择进入到哪个页面，如果没有登录则跳转到登录页面，如果已经登录则跳转到用户资料页面。在用户资料页面中，点击头像可进行头像的更改，点击修改密码跳转到修改密码页面，输入原密码与新密码后确定即可，此功能通过查找数据库来实现。点击退出登录按钮，将登录状态改为未登录。具体界面如下：

图 ‑ 个人信息与修改密码与更换头像

## 文件管理

### 文件显示

首先对检查设备内部存储中是否存在TimeTaker文件夹，如果不存在进行新建，如果果存在，则逐级读取信息按首字母排序显示在fragment\_a布局的listview中。Listview的每一个item显示的信息包括文件夹/文件图标、文件夹/文件创建时间、文件夹/文件名称。这些是靠FragmentAdapter适配器实现的，适配器中获取了相关信息，集成在file\_listview布局中。相当于fragment\_a布局的listview的每个item都使用了file\_listview布局。而文件的过滤和排序是在CustomComparator中实现的，且文件夹永远排在前面。此外，当每次对文件进行修改都会刷新界面，这点是通过改变放入FragmentAdapter适配器的信息实现的，也就是说当文件结构等发生变化，都会再次触发OpenFile方法，重新获取文件列表。具体效果如下：

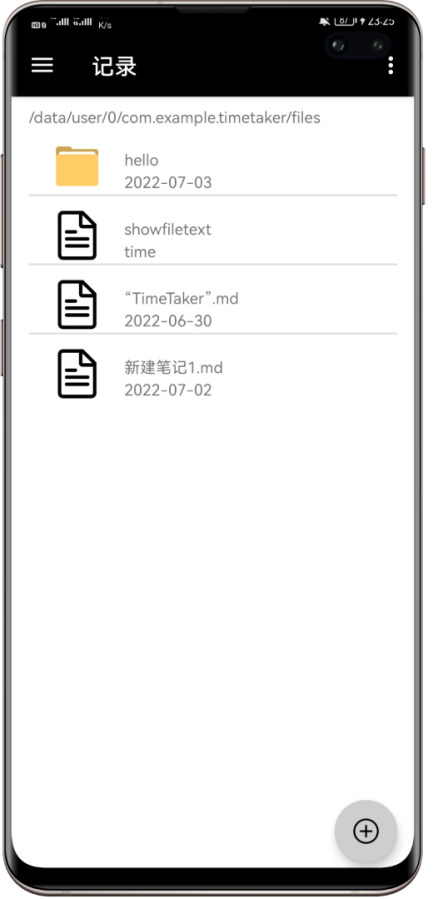


图 ‑ 文件显示

### 文件夹重命名

当用户长按选中文件或者是文件夹，会弹出文件操作菜单，再点击重命名图标即可将文件夹进行重命名。文件夹的重命名是使用了File库中的renameTo（）方法。只要用户选择该操作就会弹出AlertDialog，用户只需要在对话框中输入新名称，点击确定即可。具体界面如下：



图 六‑4对文件夹hello进行重命名

### 文件回收

当用户长按选中文件或者是文件夹，会弹出文件操作菜单，再点击回收图标即可将文件夹/文件放入回收站。这是因为当长按选中时，我们通过获取坐标得到了选中的文件，然后获取此时系统时间，将要删除的文件信息写入到本地数据库recycle.db中。信息获取完毕后，调用FileOpreationUitl文件管理工具类中的copyDirectory（）方法将文件拷贝到回收站一份，接着使用deleteFile（）方法删除当前目录下的此文件即可，也就实现了逻辑上回收，具体界面如下：

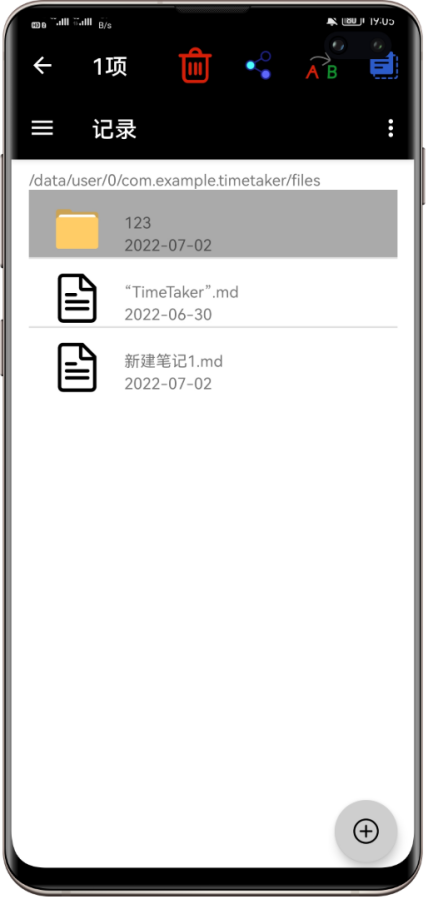


图 ‑ 删除文件夹123

### 文件移动

当用户长按选中文件或者是文件夹，会弹出文件操作菜单，再点击移动图标即可将文件夹/文件进行移动。这里的文件移动，准确的来说是复制和粘贴，这是因为选择移动文件后在重新回到文件结构中选择移动的目的地不易实现，所以我们将文件夹/文件的移动拆成两个部分。首先在获取被点击的文件信息，并将其在全局中保存，设置复制标志为true，然后到目的目录下，点击粘贴，此时会判断复制标志，只有在复制标志为true的情况下才会粘贴文件，粘贴完成后通过deleteFile（）方法删除原目录下的该文件夹/文件，具体界面为：



图 六‑6 移动文件新建笔记

## 文件编辑

文件编辑模块作为一个单独的fragment显示,通过NoteListFragment类实现, fragment的组件XML结构如下

* layout
  + data
    - variable
  + androidx.constraintlayout.widget.ConstraintLayout
    - com.google.android.material.textfield.TextInputLayout
    - com.example.timetaker.ui.edit.WYSIWYG
    - androidx.appcompat.widget.Toolbar

核心实现类及其功能如下

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 功能 |
| EditFragment | 类 | fragment的实现类,负责界面绘制与用户交互 |
| WYSIWYG | 类 | 继承自WebView的编辑器 |
| EditorViewModel | 类 | 保存数据，保存当前Note状态 |

利用Jetpack组件中的 DataBinding 与 ViewBinding实现控件与数据的绑定。

在该fragment中将笔记显示控件的内容与SimpleNote类进行关联,如下例

<data>

<variable name="note" type="com.example.timetaker.data.model.SimpleNote" />

</data>

而Markdown作为纯文本格式的标记语言,与部分html标签对应,适合进行浏览器渲染。构建WYSIWYG组件进行笔记的渲染。通过JS代码执行如下类型代码实现加粗等效果

document.execCommand('bold', false, null);

通过添加img, audio,video等标签和系统自带的图库等Intent获取文件URI实现图片与音频与视频的插入.

webview页面结构如下：

* editor.html
* normalize.css
* style.css
* w3.css
* wysiwyg.js

最终构建的界面如下

## 笔记列表

该模块作为一个单独的fragment显示,通过NoteListFragment类实现, 在NoteListViewModel类中进行核心的Note列表的获取，利用NoteListAdapter更新NoteListFragment的界面，通过实现OnNoteClickListener接口实现鼠标响应，即点击进入编辑器.

其核心实现类及其功能如下。

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 功能 |
| NoteListFragment | 类 | fragment的实现类,负责界面绘制与用户交互 |
| NoteListAdapter | 类 | 适配器类，重写方法，其对象作为recyclerView的适配器 |
| ViewHolder | 类 | 关联数据对象与行模板，实现事件响应函数 |
| NoteListViewModel | 类 | 保存数据，进行Note列表的获取 |
| OnNoteClickListener | 接口 | 响应函数的接口 |

## 设置

在主题切换部分，首先是在res/values/thems中设置两套主题“Dark”和“Light”。然后在AndroidMainfest.xml文件中将所有活动的主题设置为初始主题Light。接着在每个活动的onCreate（）方法的super.onCreate语句前进行判断，获取本地持久化信息useMyTheme，如果是Dark就使用setTheme（）方法设置主题为Dark。而主题的本地持久化，在设置中选择了主题就会实现，具体使用的方法是使用SharedPreferences（），界面如下：

图 ‑ 暗主题与亮主题

## 回收站

回收站的具体实现，其本质上是一个名叫.cr的文件夹，该文件夹存在在根目录TimeTaker下，但是在文件列表显示的时候我们并未将其显示出来。而且修改了其openFile方法，即当点击时不能进入文件夹，或者是打开文件编辑。但长按文件或文件夹，会弹出操作菜单，即可以彻底删除，或者是恢复到原目录下。

彻底删除的实现，是使用了deleteFile（）方法，改方法是对File库的文件删除的进一步封装。

恢复的实现，是因为在删除文件夹/文件时就已经将文件名，删除时间，删除前路径等信息保存在了本地sqlite数据库recycle.db中，只需要在其中查询需要恢复文件的原始路径即可。界面如下：



图 六‑8 回收站恢复与删除

## 计划

计划模块，主要用到的控件是CalendarView,对CalendarView设置点击响应，点击某一日期，如果在根目录的.cl文件夹中不存在以X年-X月-X日.md和X年-X月-X日Delete.md命名的文件则新建。所谓的添加计划其实就是在X年-X月-X日.md文件中插入一行，所谓的已完成就是将X年-X月-X日.md文件中该行删除，放到X年-X月-X日.md文件中。

这个点击和显示是通过listview控件实现的，已完成和未完成对应两个listview，同时也对应了两个适配器，每个适配器中都设置设置了Checkbox控件，并将其与listview的item点击进行绑定，从而实现无论是点击Checkbox还是item都会触发item删除操作，删除完后会刷新两个listview，具体的刷新方式同样是重新打开文件。界面如下：



图 ‑ 计划

## 帮助

帮助这个Fragment我们仅使用了一个控件WebView,WebView使用loadUrl（）方法加载团队博客，同时为了使跳转不会调用设备浏览器，使用setWebViewClient（）方法，将其返回值改为false。界面如下：



图 ‑帮助

## 分享

分享功能的实现是调用系统的分享功能，主要就是获取笔记的Uri，这里使用FileProvider库的getUriForFile（）方法，并设置文件类型为全部，再使用Intent的putExtra（）方法，启动分享活动即可。界面如下：



图 ‑分享

## 外部载入

外部载入首先需要动态获取设备的存储权限，然后使用Tintent打开对话框选择文件，接着重新onActivityResult（）方法，通过沙盒技术通过文件的Uri获取文件的实际路径，并将文件移动到TimeTaker根目录下。然后将Markdown文本使用MDTool库转换成HTML文件进行编辑，编辑完成保存再用Remark库将HTML文件转换为Markdown文本保存。该文件后续可以通过分享给第三方APP继续编辑。界面如下：

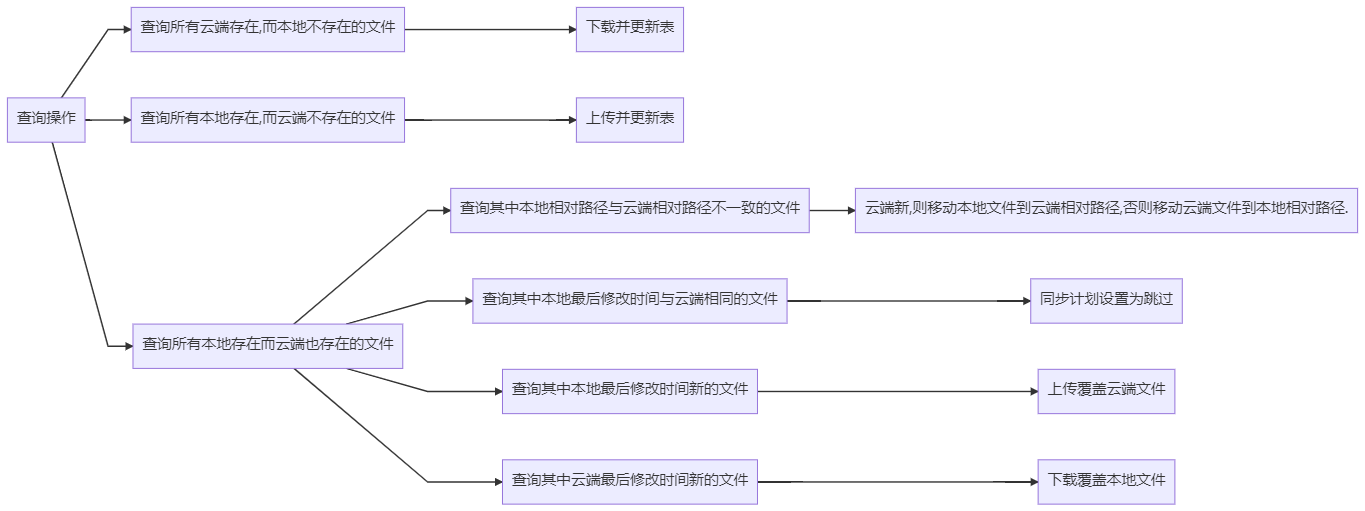
 

图 ‑ 载入与选择文件

## 云同步

本APP按照如下策略实现云同步:

* 每一个用户，在本地维护一张表，当应用打开后动态维护，目的是减少遍历文件的次数 保存：用户id,文件的本地相对路径（本地云端是一致的）(作为主码)，本地是否存在，本地最后修改时间，文件名，云端是否存在，**文件**的云端相对路径,云端最后修改时间，同步计划（上传，下载，跳过,移动,删除）
* **第一次打开应用**，扫描本地所有文件，更新表，扫描云端文件，更新表。
* **如果文件发生修改**：本地表查询该文件,更新**文件**的本地最后修改时间等
* **如果文件发生移动**:本地表查询该文件,更新**文件**的本地相对路径等
* **如果文件移入回收站**:找到该文件,更新**文件**的本地相对路径为回收站
* **点击同步**，按照如下策略开始同步



界面如下：

## 多人协同

本APP按照如下策略实现多人协同:

1.云端维护一张权限表，表名包括:用户id,公共权限文件名，权限拥有者

2.用户选择多人协同功能，并填写分享人ID，生成分享链接（可解析出用户id，文件相对路径）

3.其他用户打开分享链接，对比云端权限表中用户id，公共权限文件名的权限拥有者是否包含该用户，包含则可下载该文件

4.定时上传与下载，实现类似多人协同

界面如下：

# 测试

## 测试计划

### 编写目的

以下为TimeTaker的测试计划,阐述该项目测试资源配置、测试目标确认、详细测试策略以及其他测试活动。

### 测试策略

* 功能性测试:检查产品是否正确实现了功能,即调用该服务，是否可以将客户端文件上传到服务器端。本文中主要采用**黑盒测试**技术进行测试.
* 兼容性测试:是否能够在不同手机终端上正常运行.
* 性能测试:使用多个用户同时登陆TimeTaker，检验软件性能。
* 用户界面测试:检查界面是否美观合理，功能调整是否正常

### 测试环境

|  |  |  |
| --- | --- | --- |
| 资源名称 | 资源项 | 描述 |
| 手机移动端测试 | 操作系统 | Android 11（MIUI 12.5） |
| 手机移动端测试 | 操作系统 | Android 11（Realme UI3.0） |
| 手机移动端测试 | 操作系统 | Android 11  （鸿蒙OS 2.0） |
| 平板移动端测试 | 操作系统 | 鸿蒙OS 2.0 |

## 测试用例及其结果

### 笔记名称测试

* 笔记名称为中文**--正确**
* 笔记名称为英文**--正确**
* 笔记名称中含有特殊字符**--正确**
* 笔记名称为数字**--正确**

### 笔记内容编辑测试

* 笔记内容包含中文**--正确**
* 笔记内容包含英文**--正确**
* 笔记内容中含有特殊字符**--正确**
* 笔记内容为数字**--正确**
* 笔记内容包含图片**--正确显示**
* 笔记内容包含录音**--正确显示**
* 笔记内容包含视频**--正确显示**
* 笔记内容包含加粗，斜体，下划线**--正确显示**
* 笔记内容进行回退与前滚**--正确显示**

### 文件夹显示测试

* 新建文件夹名包含中文**--正确**
* 新建文件夹名包含英文**--正确**
* 新建文件夹名中含有特殊字符**--正确**
* 新建文件夹名为数字**--正确**
* 2层，3层，4层嵌套文件夹—**正确显示**
* 文件的移动，复制，剪切—**正确**
* 文件移动到回收站再恢复--**正确**
* 文件移动到回收站再恢复再删除--**正确**
* 文件移动到回收站再彻底删除--**正确**
* 文件分享—**正确**
* 笔记列表视图—**正确显示**

### 计划测试

* 当前日添加事项—正确
* 次日添加事项—正确
* 前一日添加事项—正确
* 添加连续10日的事项—正确
* 将当日未完成事项改为已完成—正确
* 将当日已完成事项改为未完成—正确

### 云同步与多人协同测试

* 本地一个笔记，同步所有笔记—云端数据正确，同步计划编排正确
* 本地包含嵌套文件夹笔记，同步所有笔记—云端数据正确，同步计划编排正确
* 本地一个笔记，同步所有笔记—云端数据正确，同步计划编排正确
* 未登陆进行同步—显示请登陆，正确
* 本地没有笔记，云端有笔记，同步所有笔记—云端数据正确，同步计划编排正确，上传下载正确。
* 本地没有笔记，云端有嵌套文件夹笔记，同步所有笔记—云端数据正确，同步计划编排正确，上传下载正确。
* 本地一个笔记，同步所有笔记后再次更新本地笔记—云端数据正确，同步计划编排正确
* 本地一个笔记，同步所有笔记后再次删除本地笔记到回收站—云端数据正确，同步计划编排正确
* 本次创建连续3日的计划事项--云端数据正确，同步计划编排正确
* 生成分享链接—正确
* 打开分享链接—正确

### 界面测试

* 按钮文字是否正确--**正确**
* 提示的文字是否正确--**正确**
* 侧边栏跳转**—正确**
* 侧边栏多次跳转后选择返回键—提示连续两次返回退出，正确
* 侧边栏多次跳转后选择返回键两次—退出程序—正确
* 文件夹结构到笔记视图到文件夹结构—正确
* 笔记视图调整笔记编辑—正确
* 文件夹结构调整笔记编辑—正确

### 兼容性测试

* 手机端Android 11（MIUI 12.5），Android 11（Realme UI3.0），鸿蒙OS 2.0测试正常
* 平板端鸿蒙OS 2.0测试正常

### 其他功能测试

* 同步过程中断网--**断网时仍然保持上传中状态,网络恢复后自动同步**
* 同步过程中服务器突然停止工作—**同步失败**
* 同一用户登录多个客户端—正常使用
* 同一用户登录多个客户端，并进行云同步—正常使用

## 测试结果分析

* 功能性测试:总体功能可以正常实现，即便用户出错，程序可以进行明确的提示并且系统不会因此而崩溃.
* 兼容性测试:能够在Android与鸿蒙系统下的手机与平板运行
* 用户界面测试:界面较为美观合理,系统交互正常.

# 总结

## 开发难点

### 各Fragment之间切换、返回逻辑的设计

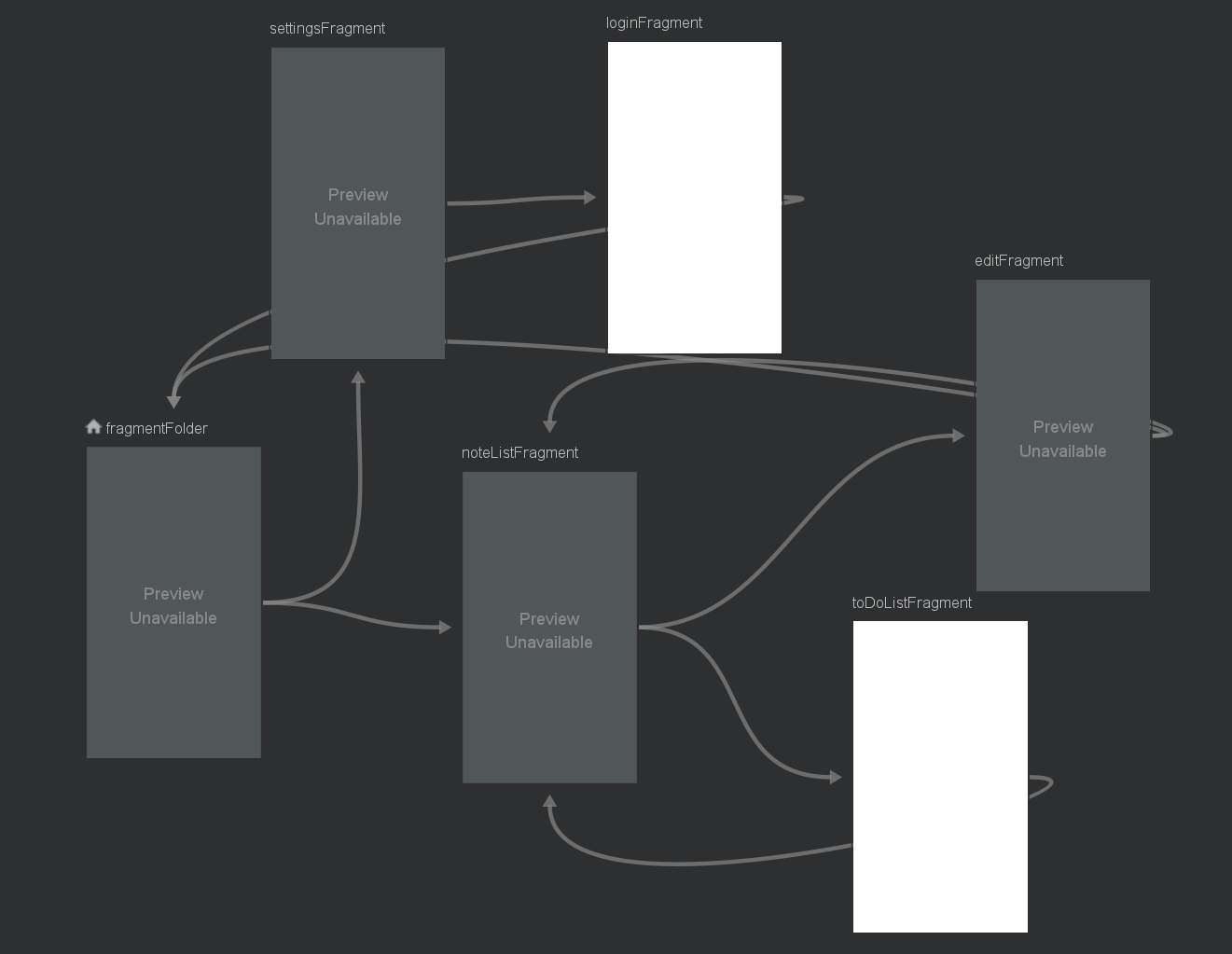


图 八‑1导航图

我们使用导航图将Fragment接入FragmentController,这样我们点击侧边栏菜单就能进入一个新的Fragment，但是随之而来的是返回问题，因为多Fragment实际上是一个Activity，点击设备自带的返回 按钮相当于杀死一个活动，也就是说每次进入一个Fragment就不能返回，否则直接退出程序，这是不符合逻辑的。我们设想的是，在主界面，点击一次返回提示“再次点击退出”，第二次点击返回才会退出程序。而在每一个Fragment中，点击返回都会弹出侧边栏，可以重新进行选择，如果意向退出，只需再次点击返回即可。当我们实现这些以后，突然发现，如果在一个Fragment中点击返回，然后进入到另一个Fragment中再次点击退出，程序没有按我们料想的弹出侧边栏，反而直接退出了，为此我们想到了记录下上次的Fragment信息，比较两次Fragment是否相同，来执行不同的返回逻辑。这一切解决后，我们在测试时又发现，如果在某一Fragment中点击返回，然后再点击设备右侧空白区收回侧边栏，而不是再次进入这个Fragment，会出现文件列表View和该Fragment View相互重叠的Bug，于是我们设置了第二个标识符，进行逻辑辅助判断，最终解决了各Fragment之间切换、返回逻辑设计的难题。

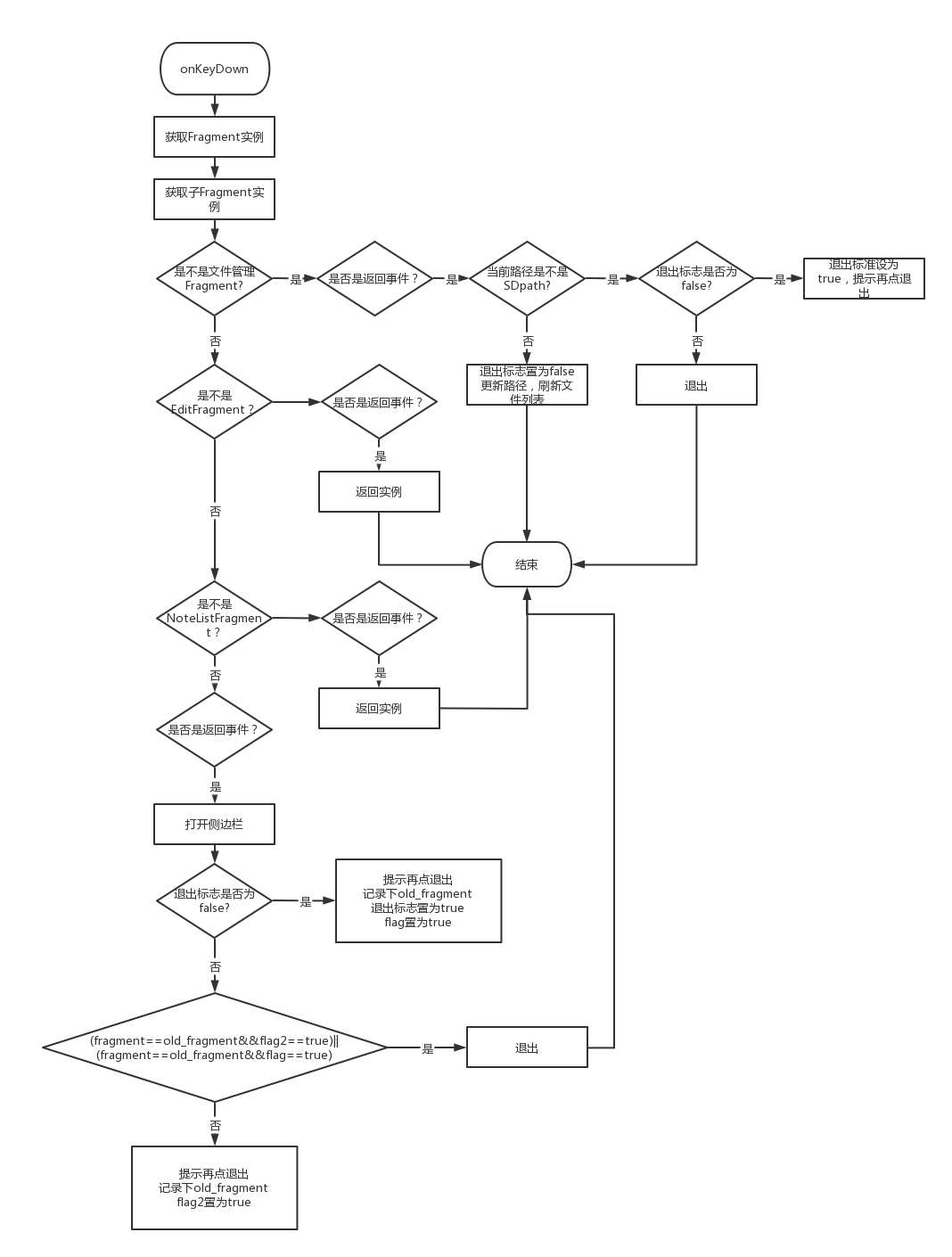


图 ‑返回逻辑设计

### Android与HarmonyOS之间的不兼容

在APP的开发过程中，我们前后两次遇到了系统不兼容的问题。第一次是在文件管理部分，我们希望在手机内存中新建根目录，但是却发现在Android设备上无问题，在HarmonyOS系统上闪退。通过查阅资料我们发现，HarmonyOS系统对于APP访问外部储存有着诸多限制,最终我们在软件中添加可供选择为外部存储与内部存储,而默认为内部存储以解决此问题.

第二次是我们在Light主题中多设置了一个属性android:top，导致APP运行在HarmonyOS系统上，长按任何编辑框都会闪退。因为是在xml文件里写的，所以程序并没有报错，抛出的异常在网络上也搜不到，没有参考。后来我们通过二分法，定位才发现是这里不兼容导致了冲突的问题。

### App对设备权限的动态获取

在APP的开发过程中我们遇到了无法访问设备相机的问题，最初认为是自己的代码问题，调试后发现，是权限没有开放。原来Android6.0以前的系统（API < 23）采用的这种方式，只要用户在AndroidManifest.xml中注册了权限，安装APP后默认就获取了这些权限。但随着系统的升级，Google也意识到静态申请权限的弊端，所以在Android6.0中，对权限进行了重新梳理，将权限分为普通权限和危险权限：正常权限：不会给用户隐私带来危险的权限，只要开发者在AndroidManifest.xml中注册了，系统将自动授权；危险权限：可以访问用户隐私数据的权限，必须获取用户的同意才可获得授权，而相机的使用就属于危险权限，所以在AndroidManifest.xml中注册无效，还要进行动态申请，具体的方法是先通过checkSelfPermission（）方法判断是否具有该权限，如果没有再通过requestPermissions（）方法弹出对话框，让用户选择授权。

### Android studio连接云MySQL数据库

此模块遇到的困难主要是连接云数据库，在Android studio上连接数据库之前先用Java测试了一下，在Java上没问题，结果到了Android studio上却连接不上。经过查阅资料得知要开一个子线程才能连接数据库，同时也要在AndroidManifest.xml中获取网络权限。开子线程是因为一个APP如果在主线程中请求网络操作，将会抛出NetworkOnMainThreadException异常。Android这个设计是为了防止网络请求时间过长而导致界面假死的情况发生，因此要在子线程中连接数据库。

## 协作问题

本次APP开发是团队合作，为了能够及时更新代码，保存记录回滚，我们采用Git进行版本控制。同时通过Sourcetree进行拉取、上传、合并。

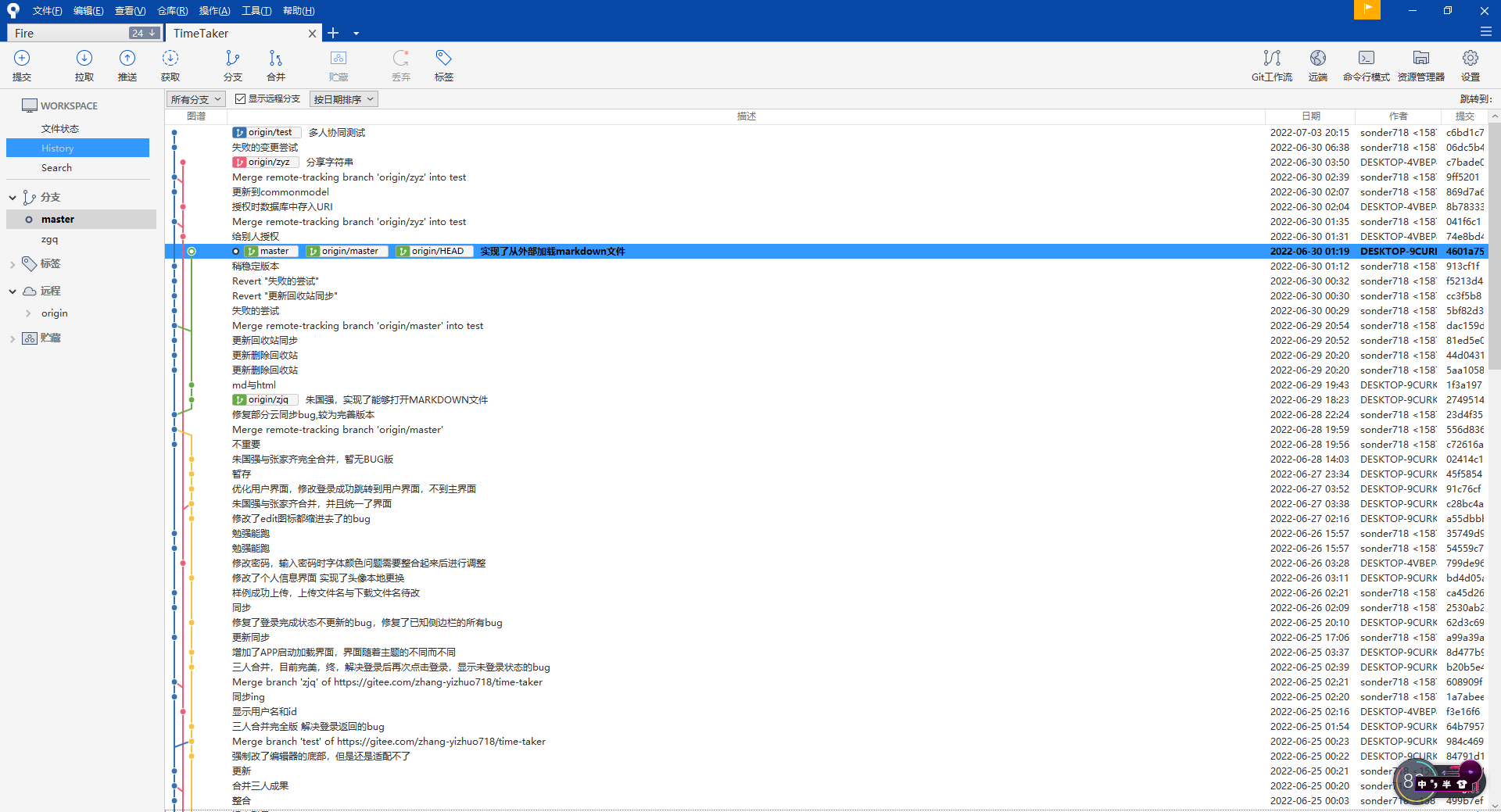


图 ‑协作记录

## 经验教训

行笔至此，终于要为两个月的云记事本开发画上一个句号。

回首看，从选题、设计、开发再到测试，一路走来我们所遇困难重重。团队共有的困难是没有过Android开发经验，从零开始学习Android的活动与布局。其次是编程语言遇到的困难，团队中没有人接触过Kotlin,更没有人接触过后端开发，只有一人学习过Java。刚开始的时候我们买了“第一行代码这本书”去学习，结果半个月过去，项目毫无进展，最后我们选择一边推进，一边学习的方式开始开发，并通过不断下载编译开源代码来学习具体功能实现的方式。结果这是一条正确的路，我们顺利完成了第一次中期汇报，虽然那个时候成果很少，也不知道能走多远，但事情都在向好的方向发展。所以我们的第一个收获是学习思路的转变，作为科班出身的程序员，学习是本能，学院和老师不肯能教会我们所有东西，我们学到的是自主学习的能力，在困难面前保持自信，科学分析，理清思路，一定可以取得关键性的突破。

在一个项目的开发过程中，我们不能采用串行模式，必须并行推进，那么如何整合代码，如何进行版本控制就成了一个难题。虽然这不是我们第一次使用Git，但却是使用最久的一次，一次次的推送失败，合并错误，我们在摸索中前进，并结合AndroidStudio的history功能，成功将合并时的冲突降到了最低。所以，较为熟练的使用Git和Sourcetree是我们的第二个重要收获。

第三个收获，来自于如何进行团队分工与合作。我们团队来自同一个班级，同一个寝室。所以相比于其它团队，我们的优势在于可以面对面交流，互相鼓励。在本次开发过程中，张一卓作为队长把握着开发方向，提供了很多宝贵的思路；朱国强则像裁缝一样，到处修修补补，调试总体上的Bug；张家齐则像探测器一样，不断寻找可以完善的地点。三人明确分工，又相互合作，所以我们最终产品和最初设计出入少之又少。

除了收获，我们还有惨痛的教训。第一个是团队成员没有及时的上传本地代码，拉取最新项目。这就导致团队成员的本地版本有着很大出入，在合并的时候存在很多冲突。第二个是没有提前约定好命名规范，导致文件命名五花八们，在其它成员阅读代码时需要不断的搜索，交流。第三个是没有做到处处打log，这导致我们的代码出现了bug就只能凭借着感觉不断的缩小报错的范围去寻找出错点，大大降低了开发效率。

虽然一路坎坎坷坷，但我们还是充满了成功的喜悦。首先感谢杨鸣老师的陪伴，每个周五的晚上，风里雨里都守在机房，并在四次汇报中不断为我们提出宝贵的修改建议。其次要感谢团队的每一个成员，在这个开发过程中，我们互相督促，很多次看到了凌晨马路上昏黄的灯光。“我累了，休息会儿”，“起来啊，干活”，“接着肝”成了这个寝室、这个团队最常听到的声音。如果没有每一个成员的坚持，我们的项目最后一定烂尾。

“追风赶月莫停留，平芜尽处是春山”这是我们第一次进行前端加后端的完整项目开发。也感受到系统能力培养这门课综合了我们所有所学知识，实实在在地提高了我们的能力。第二次跟老师说再见，祝老师工作顺利，生活幸福。星光不负赶路人，也愿我们团队的每一位，前路漫漫亦灿灿！

# 参考文献

[1] <https://www.zhihu.com/question/20784991>

[2] <https://blog.csdn.net/sun_cui_hua/article/details/75648107>

[3] <https://blog.csdn.net/weixin_34352449/article/details/86023696>

[4] <https://zhuanlan.zhihu.com/p/59467370>

[5] <https://blog.csdn.net/flyersboy/article/details/118230321>

[6] <https://blog.csdn.net/qq_41241926/article/details/104872988>

[7] <https://blog.csdn.net/qq_33209777/article/details/104767631>

[8] <https://blog.csdn.net/qq_40205116/article/details/88418781>

[9] <https://blog.csdn.net/m0_56348460/article/details/122912580>

# 附录

1. **个人简历**

朱国强，男，2000年生，中共党员，就读于中国地质大学（武汉）计算机科学与技术专业，专业排名32/137，36门课程90+。曾任计算机学院团委组织部部长、计算机学院团委副书记，现任191191班团支书。持软著《AR 伴你游”智慧文旅 AR 系统》一份，荣获中国地质大学（武汉）校长奖学金，第七届“互联网+”大赛省铜，计算机学院“计科标杆”，优秀共青团干部，百名好支书等诸多荣誉。

张家齐，男，汉族，2001年生，共青团团员，2019年入学中国地质大学（武汉），现就读于计算机学院计算机科学与技术专业。



张一卓，男，2001年生，中共党员，就读于中国地质大学（武汉）计算机科学与技术专业，专业排名20/137，曾获“校级优秀共青团员”等称号。

1. **小组分工合作**
2. **课程论文写作情况**

（附录里要写，个人简历，小组分工合作，课程论文写作情况）

课程论文能够清晰描述软件设计思想、过程，描述软件结构、模块、接口，软件开发技术难点，以及本次软件开发经验教训等内容。

论文必须做到：结构合理、逻辑清晰、图文并茂、表述清楚、符合软件开发相关规范等。

（4）软件包：包括源工程文件包、可安装运行的软件系统、用户手册（运行环境说明以及软件使用说明）。