

**系统能力培养课程**

**中期报告**

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 5 月

**目录**

[一、摘要 2](#_Toc9693)

[二、概要设计 2](#_Toc5624)

[1、开发环境 2](#_Toc21575)

[2、数据结构 2](#_Toc24681)

[3、模块设计 4](#_Toc19972)

[4、接口设计 4](#_Toc22946)

[5、ER图 5](#_Toc23080)

[6、用例图 5](#_Toc19561)

[7、类图 5](#_Toc16476)

[8、流程图 6](#_Toc20949)

[三、详细设计 7](#_Toc11554)

[1、文件转发 7](#_Toc9469)

[2、文件管理功能 8](#_Toc13651)

[3、笔记列表 10](#_Toc7442)

[4、Markdown编辑器 12](#_Toc23108)

[5、登录注册 13](#_Toc18766)

[四、技术难点实验 15](#_Toc15489)

[1、设计思想的重难点 15](#_Toc17762)

[五、系统原型（设计） 17](#_Toc22787)

[六、后期工作计划 20](#_Toc21061)

[1、后期工作 20](#_Toc19772)

[2、后期分工 20](#_Toc24215)

[3、日程安排 20](#_Toc16503)

1. **摘要**

TimeTaker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，云端备份永久存储。经需求分析到设计开发，项目目前已如期实现了登录注册、笔记编辑、文件管理、笔记分享等功能，并将按照开发计划，继续完善后续功能，最终实现交付。

**二、概要设计**

**1、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+

1. **数据结构**

(1)字符串

<string name="app\_name">TimeTaker</string>

<string name="navigation\_drawer\_open">Open navigation drawer</string>

<string name="navigation\_drawer\_close">Close navigation drawer</string>

<string name="nav\_header\_title">美好时光与你环环相扣</string>

<string name="nav\_header\_subtitle">android.studio@android.com</string>

<string name="nav\_header\_desc">Navigation header</string>

<string name="action\_settings">Settings</string>

<string name="menu\_home">记录</string>

<string name="settings">设置</string>

<string name="recycle">回收</string>

<string name="help">帮助</string>

<string name="menu\_gallery">Gallery</string>

<string name="menu\_slideshow">Slideshow</string>

<string name="action\_Remove">删除</string>

<string name="action\_Copy">复制</string>

<string name="action\_Paste">粘贴</string>

<string name="action\_search">搜索</string>

<string name="drawer\_open">分类浏览</string>

<string name="action\_new">新建</string>

<string name="action\_Rename">重命名</string>

<string name="hello\_blank\_fragment">Hello blank fragment</string>

<string name="editor\_hint\_content">请输入内容</string>

<string name="edit\_text\_title">标题</string>

<string name="editor\_hint\_title">标题</string>

<string name="action\_add\_task">action\_add\_task</string>

<string name="action\_insert\_bold\_text">粗体</string>

<string name="action\_change\_notelist">笔记</string>

<string name="action\_Share">分享</string>

(2)类

**a、Package\_edit**

class EditFragment : Fragment()

class EditFragmentQuill : Fragment()

class EditorViewModel : ViewModel()

**b、Package\_foldermanager**

public class CustomComparator implements Comparator<File>

public class CustomFileFilter implements FileFilter

public class FileOpreationUitl

public class FileSort

public class FragmentA extends Fragment implements AdapterView.OnItemClickListener

public class FragmentAdapter extends BaseAdapter

**c、Package\_gallery**

public class GalleryFragment extends Fragment

public class GalleryViewModel extends ViewModel

**d、Package\_home**

public class HomeFragment extends Fragment

public class HomeViewModel extends ViewModel

**e、Package\_list**

class NoteListAdapter (private val notelist: MutableList<SimpleNote>,val listener:OnNoteClickListener) :RecyclerView.Adapter<NoteListAdapter.ViewHolder>()

class NoteListFragment : Fragment(),OnNoteClickListener

class NoteListViewModel : ViewModel()

**f、Package\_slideshow**

public class SlideshowFragment extends Fragment

public class SlideshowViewModel extends ViewModel

**g、Package\_utils**

fun View.liftAppBarOnScroll(appBar: AppBarLayout,elevation: Float)

**h、others**

class BlankFragment\_Test : Fragment()

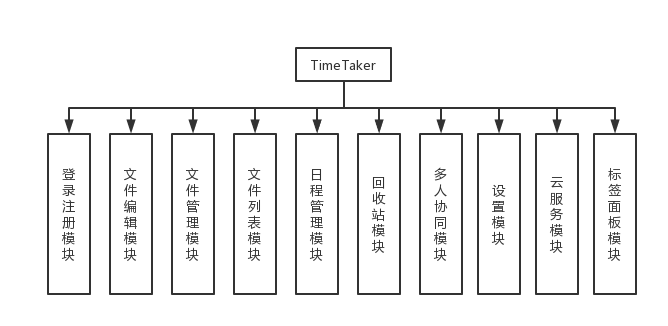
public class BlankFragmentTestViewModel extends ViewModel

public class Const

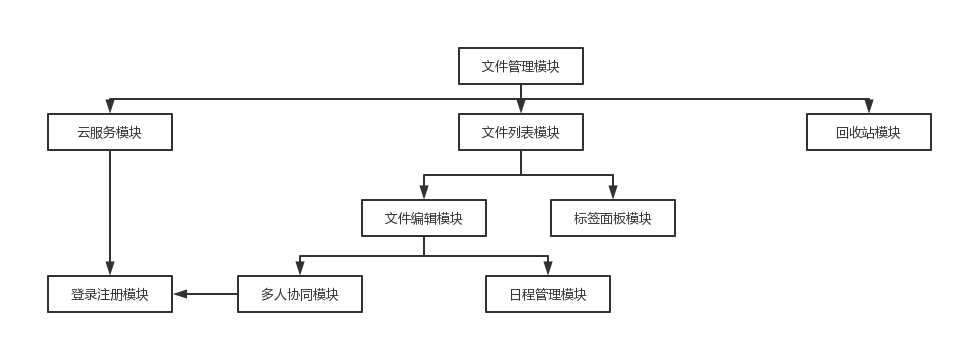
public class DBHelper extends SQLiteOpenHelper

public class Main2Activity extends AppCompatActivity

1. **模块设计**
2. 名称
3. 登录注册模块
4. 文件编辑模块
5. 文件管理模块
6. 日程管理模块
7. 文件列表模块
8. 多人协同模块
9. 设置模块
10. 回收站模块
11. 云服务模块
12. 标签面板模块
13. 模块关系



（图2.1 十大模块）



（图2.2 模块调用关系）

1. **接口设计**

**新建：**action\_new

**查找：**action\_search

**移动：**action\_Copy、action\_Paste

**删除：**action\_Remove

**重命名：**action\_Rename

**分享：**action\_Share

**上传：**action\_Upload

**登录：**action\_Login

**注册：**action\_Register

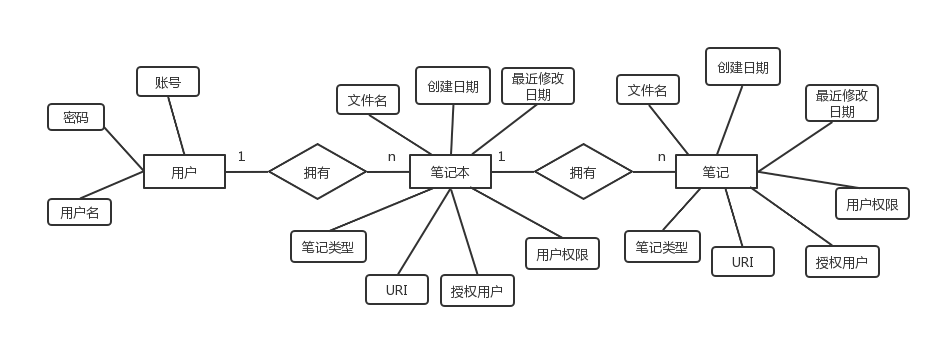
**设置：**action\_Settings

**编辑：**action\_Edit

**协同：**action\_Cooperate

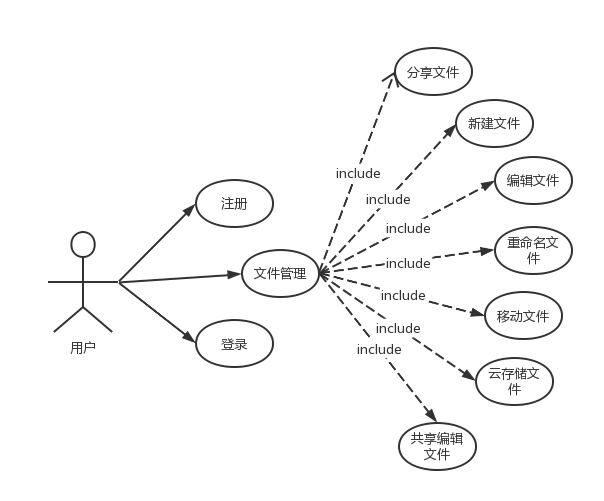
**垃圾站：**action\_Recycle

1. **ER图**



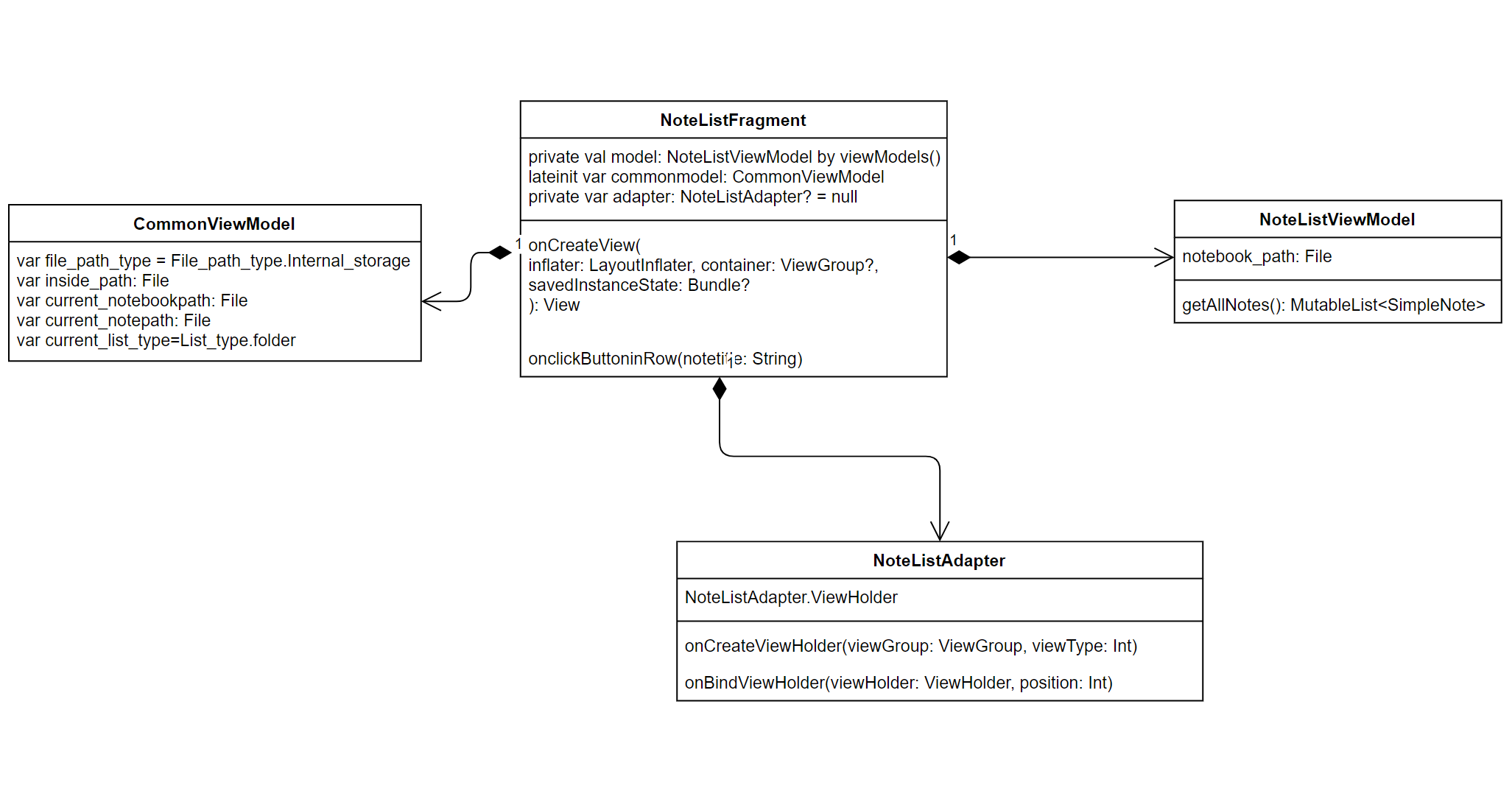
（图2.3 ER图）

1. **用例图**

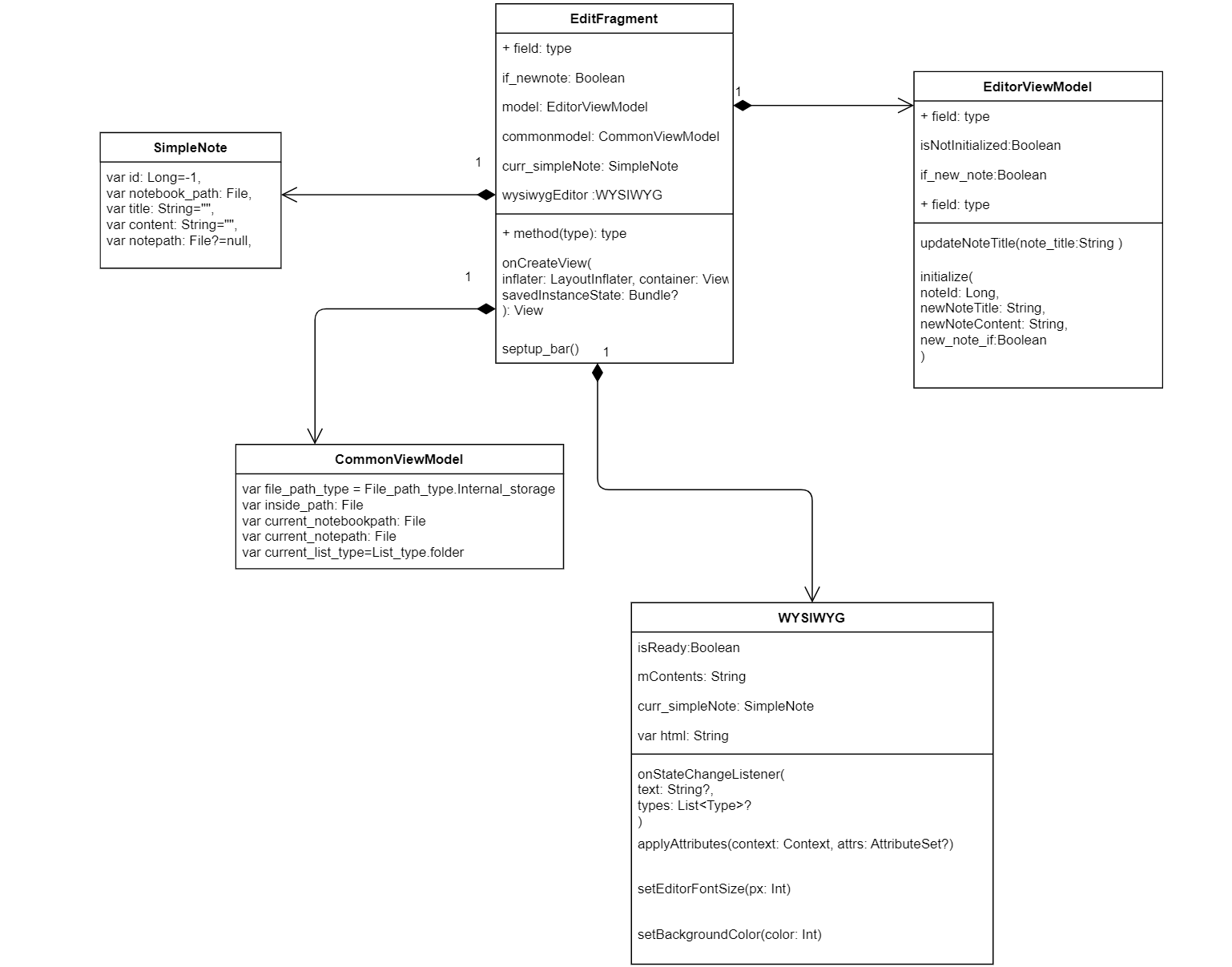
****

（图2.4 用例图）

1. **类图**

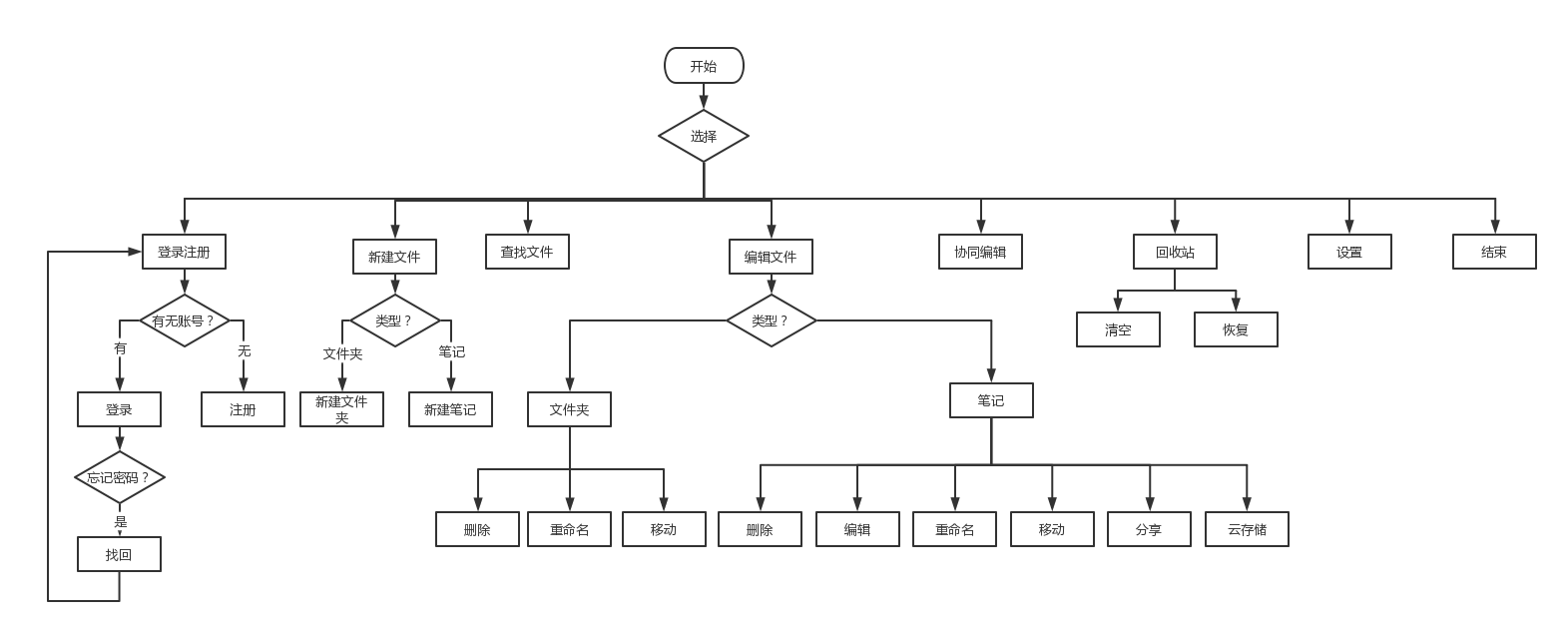


（图2.5 文件列表类图）



（图2.6 编辑器类图）

1. **流程图**



（图2.7 程序流程图）

**三、详细设计**

**1、文件转发**

（1）功能设计

长按选中文件，点击分享按钮，选中分享的手机APP，即可把笔记分享到该APP指定好友处，例如微信好友、QQ好友。

（2）界面设计



（图3.1 文件分享）

（3）功能实现概述

a、在项目res文件夹下建立xml文件夹

b、放入provider\_paths.xml

*<?***xml version="1.0" encoding="utf-8"***?>* <**paths xmlns:android="http://schemas.android.com/apk/res/android"**>  
 <**external-path name="external\_files" path="."**/>  
 <**root-path name="root\_path" path="."**/>  
 </**paths**>

这里的意思指的是文件加载路径。

c、布局中添加分享布局ic\_action\_share.xml

d、在action\_menu.xml菜单中加入分享菜单

e、在FragmentA中，添加响应方法,当点击分享菜单触发分享操作

**case** R.id.***action\_Share***:  
 **for**(**int** i=0;i<array.size();i++){  
 **if**(array.valueAt(i)==**true**){  
 **int** x=array.keyAt(i);  
 Intent intent = **new** Intent(Intent.***ACTION\_SEND***);  
 Uri uri=FileProvider.getUriForFile(getContext(),getContext().getPackageName()+ **".provider"**, **mfileData**[x]);  
 intent.setType(**"\*/\*"**);  
 intent.putExtra(Intent.***EXTRA\_STREAM***, uri);  
 startActivity(intent);  
 }  
 }  
 **return true**;

（4）核心功能实现

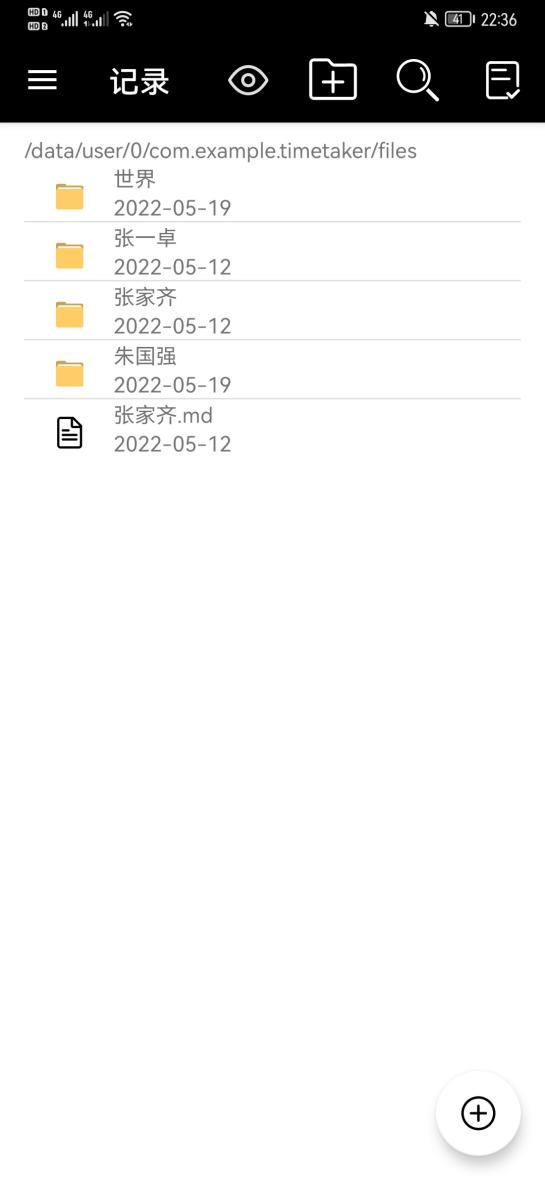
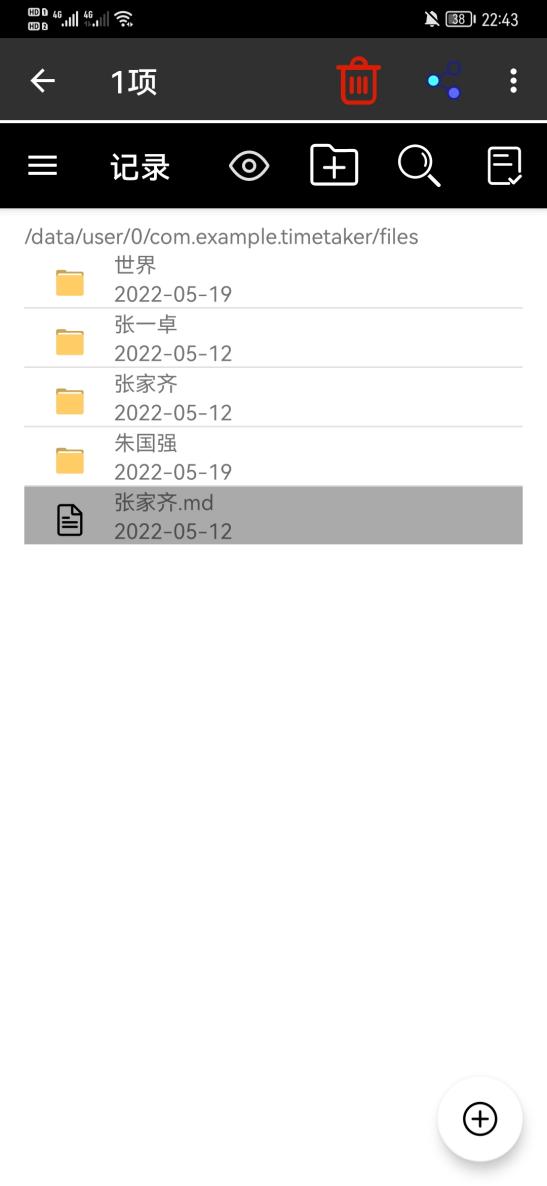
要实现Android的分享功能有两种方法，一种是接入ShareSDK,一种是原生分享。我们选择使用Intent调用andoird原生的分享功能，这种方法更加方便快捷。

1. **文件管理功能**

（1）功能设计

文件管理主要包括的文件夹的建立，文件夹的重命名，文件夹的删除，文件的重命名、删除、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。

（2）界面设计

（图3.2 文件管理）

（3）功能实现概述

实现这些功能前，需要自定义几个工具类

CustomComparator：对需要显示的文件夹和文件进行比较，按字母升序排列，将该方法传递给sort方法，从而允许在排序顺序上实现精确控制。

FileSort:对文件列表显示顺序进行确定，让文件以名称排序的方式罗列出来。

CustomFileFilter:抽象路径名的过滤器，测试指定的抽象路径名是否应该包含在路径名列表中。如果不是文件就返回。

FileOpreationUitl：文件操作的具体实现，包含文件的删除、文件复制、文件夹复制、新建文件。

FragmentAdapter：Fragment适配器，在这里实现了每个Fragment应该显示什么，如何显示。

（4）核心功能实现

接下来是文件管理功能实现的具体Fragment,FragmentA。

a、绑定数据：判断SD卡是否正常，如果SD卡正常，读取指定文件夹TimeTaker中的内容，如果是初次安装APP，则自动在内部存储中生成空文件夹，该文件夹就是所有笔记存放的位置。

b、获取文件：使用工具类过滤.文件，并对当前路径下的所有文件夹、文件进行排序。

c、长按选中：使用CHOICE\_MODE\_MULTIPLE\_MODAL进入ListView的多选模式。进入多选模式后长按item、点击item可以实现选中和不选中的切换，同时在多选模式下如果当前选中item的个数为0，也不会退出多选模式；在选中状态下可以通过手机系统的返回键回到普通模式；在普通模式下点击item则弹出文本内容；普通模式下长按item再次进入多选模式。

d、文件夹重命名：在选中状态下点击菜单重命名项，弹出AlertDialog框输入新名称即可。具体实现的方式是在获取到该文件夹所有信息后使用包里的renameTo方法，这样对文件夹重命名后文件夹内部的文件都还存在，如果使用删除重建，就要把所有的文件保留下来，在文件夹新建好后移动进去，这个过程比较繁琐也比较浪费资源。

e、文件夹/文件的删除：在选中状态下点击菜单删除项，调用FileOpreationUitl工具类中的deleteFile方法即可实现。

f、文件夹/文件的移动：所谓的移动就是选中一个文件夹或者是文件，然后回到目录下，选择位置放入。这个过程看似简单，实现起来却比较麻烦。主要涉及到FragmentA和HomeFragment的交互。我们采用的思路是将文件夹/文件的移动拆解为两个动作，第一个动作是复制，并记录下该动作，然后跳转到HomeFragment点击菜单粘贴项进行粘贴，一旦粘贴完成，将复制标记取消，删除原路径下的文件夹/文件。在复制未被标记的情况下，选择粘贴不会触发任何操作。

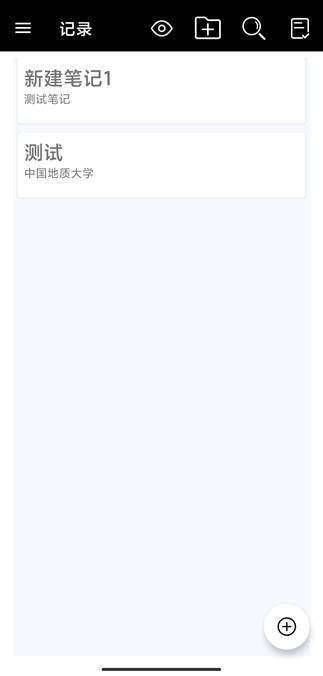
g、新建文件夹：在设计初期，我们希望打开APP就能看到自己的文件管理系统，所以新建文件夹的实现位于HomeFragment内，选择菜单新建文件夹项，就可以在当前路径下创建新的文件夹。

1. **笔记列表**

（1）功能设计

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

（2）界面设计



（图3.3 笔记列表）

其组件树结构如下

* androidx.constraintlayout.widget.ConstraintLayout
  + androidx.recyclerview.widget.RecyclerView

（3）功能实现概述

该模块作为一个单独的fragment显示,通过NoteListFragment类实现,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 |

（4）核心功能实现

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

1. **Markdown编辑器**

（1）功能设计

本模块的核心功能是渲染打开的Markdown文件，并能够进行各种形式的编辑，如加粗，添加图片，添加录音等。

（2）界面设计

功能界面由标题栏，内容栏，底部编辑功能栏，顶部功能栏组成。

各功能组件包含以下功能：

|  |  |
| --- | --- |
| **组件名称** | **功能** |
| 标题栏 | 输入与显示标题 |
| 内容栏 | 输入与显示经过Markdown渲染后的内容（包括文字，图片，音频） |
| 底部编辑功能栏 | 对内容进行操作，功能包括：加粗，斜体，高亮，一级标题，二级标题，添加图片，添加音频等 |
| 顶部功能栏 | 对笔记进行操作，功能包括：导出，分享，搜索，协同编辑，切换编辑/查看视图等 |



（图3.4 编辑器）

其组件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

（3）功能实现概述

该模块作为一个单独的fragment显示,通过NoteListFragment类实现,其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 |

（4）核心功能实现

a、数据绑定

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

<data>

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

</data>

\b、webview编辑器

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。

webview页面结构如下：

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

1. **登录注册**
2. 功能设计

实现用户的登录、注册以及阅读用户协议。

1. 界面设计

（图3.5 登录注册）

1. 功能实现概述

a、连接数据库

使用jdbc连接到云端数据库。

b、登录功能实现

①使用TextUtils.isEmpty()判断EditText里面的账号和密码是否为空，若为空，则用Toast弹出消息进行提示；若不为空，则对数据库中UserInfo这个存储用户信息的表进行遍历，看有没有与输入的账号密码相匹配的，若找到匹配的，则跳转到主界面，若没有匹配的，则用Toast弹出消息提示账号或密码错误。

②输入账号这一个EditText里面，点击下拉的标志显示最近登陆过的账号，用列表存储。

③点击叉号清除掉输入的账号或者密码。

④点击密码栏的眼睛图标来显示或者隐藏密码。

⑤点击忘记密码用来找回密码。

⑥点击注册账号进入账号注册页面。

⑦点击服务条款阅读本软件的用户须知。

c、注册功能实现

①用户输入自己的账号和密码，并且确认密码。

②将用户的账号与密码存储到数据库中。

③注册成功后返回登录界面。

d、用户须知功能实现

点击该按钮之后进入阅读用户协议页面，用户选择是否同意该协议，同意之后方可继续使用本软件。后期根据相关法律法规来确定该界面具体内容。

1. 核心功能实现

使用st.executeQuery()函数和st.setString(1, id\_)函数读取pg数据库中UserInfo表中的信息，在遍历过程中如果EditText中输入的id和password与数据库中的id和password相同则登陆成功，跳转到主界面；若不相同则用Toast显示账号或密码错误的消息。

将Toast封装，使其可以在子线程中显示消息。初始化Toast之前先判断当前线程的looper是否为空，为空则初始化一个新的myLooper，然后在调用Toast的show方法之后让looper启动起来即可。因为Looper的loop()方法是无限循环的，为了防止Looper阻塞线程，导致内存泄漏应该及时退出Looper。

**四、技术难点实验**

**1、设计思想的重难点**

程序的设计思想如下：

1. MVVM现代架构设计：严格的MVVM架构，具有较强可扩展性
2. 纯Markdown格式笔记文件：便于用户自定义与文件的批量转移。
3. 文件夹组织形式：笔记通过文件夹进行组织,便于归类整理。
4. 数据库存储笔记路径：数据库中存储笔记路径与配置、权限等信息
5. WebView组件实现Markdown编辑器：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

**2、MVVM架构**

MVVM是Model-View-ViewModel的简写，其中

View：对应于Activity和XML，负责View的绘制以及与用户交互。

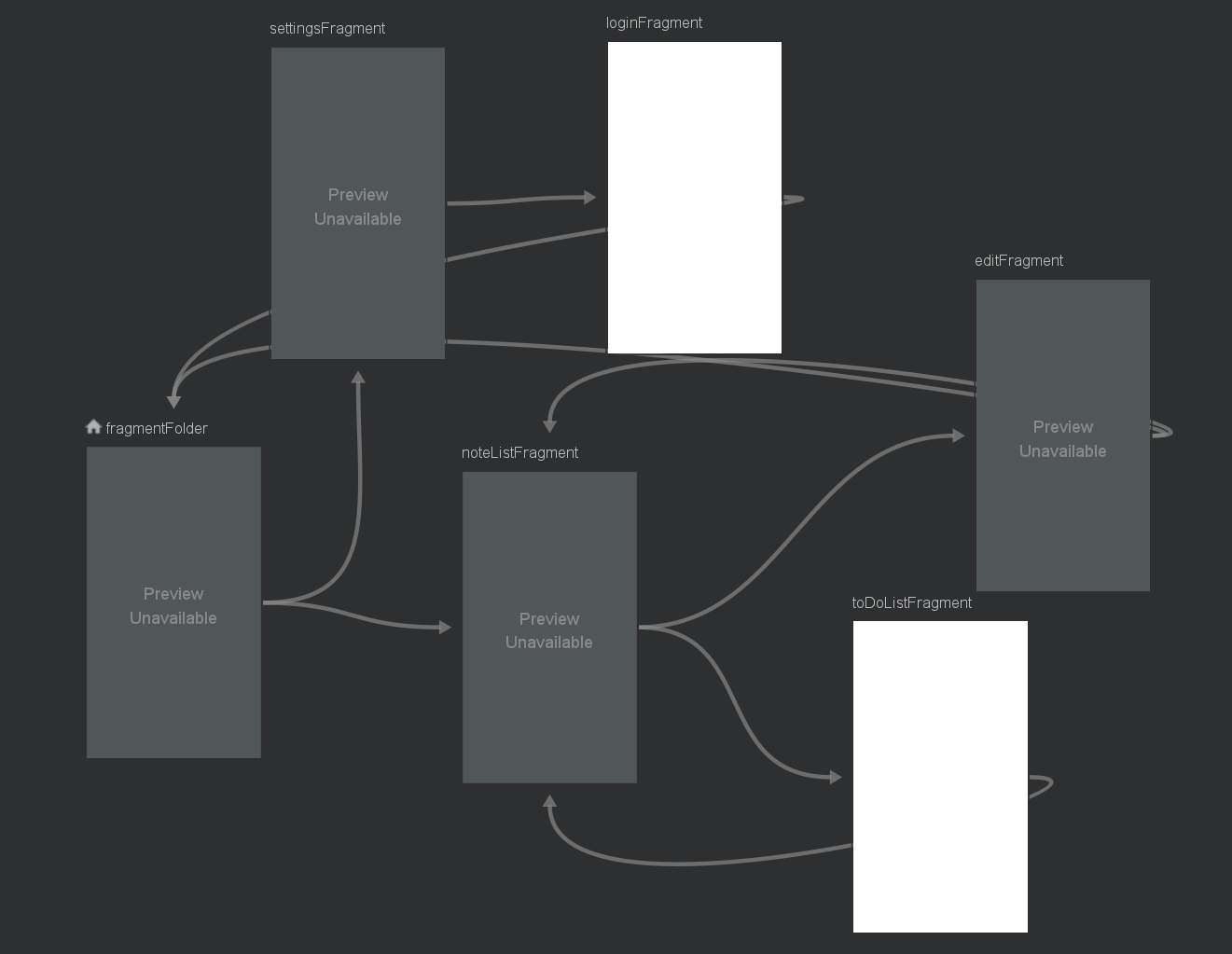
Model：实体模型。

ViewModel: 负责完成View与Model间的交互，负责业务逻辑

（1）View层

采用单activity多fragment的页面组织结构，使用Jetpack组件中的Navigation组件对fragment进行管理与跳转，利用Jetpack组件中的DataBinding 与 ViewBinding实现控件与数据的绑定。

多fragment的导航图如下：

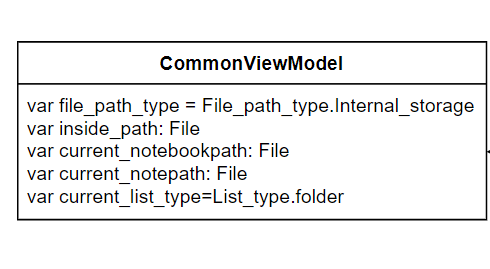


（图4.1 导航图）

（2）ViewModel层

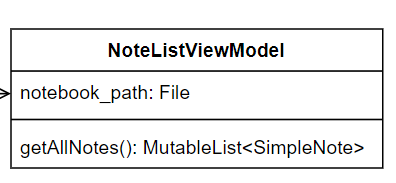
ViewModel只做和业务逻辑和业务数据相关的事,通过Jetpack组件中的ViewModel与LiveData实现

建立CommonViewModel进行各fragment间的数据交流,存储如当前笔记路径,当前笔记名称等,当前打开笔记本名称路路径等信息

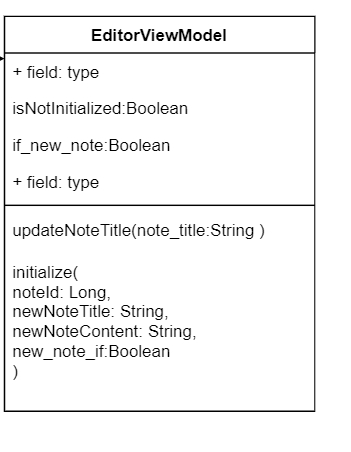


（图4.2 ViewModel层）

各个fragment建立各自的ViewModel,由于其独立activity生命周期,避免数据受到屏幕旋转等生命周期变化的影响. 其只关注数据和业务逻辑,不涉及UI控件的引用与更新,实现View和ViewModel之间的松散耦合.



（图4.3 NoteListViewModel）

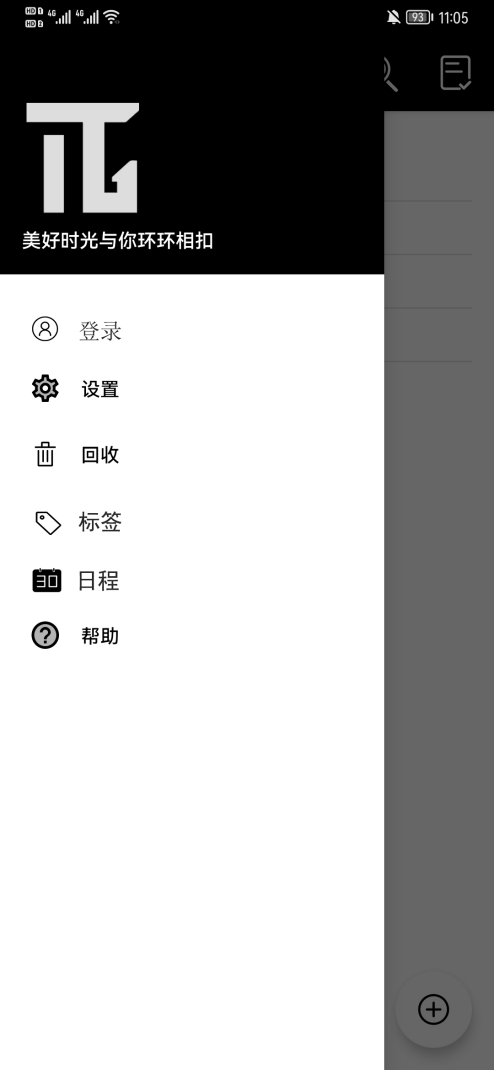


（图4.4 EditorViewModel）

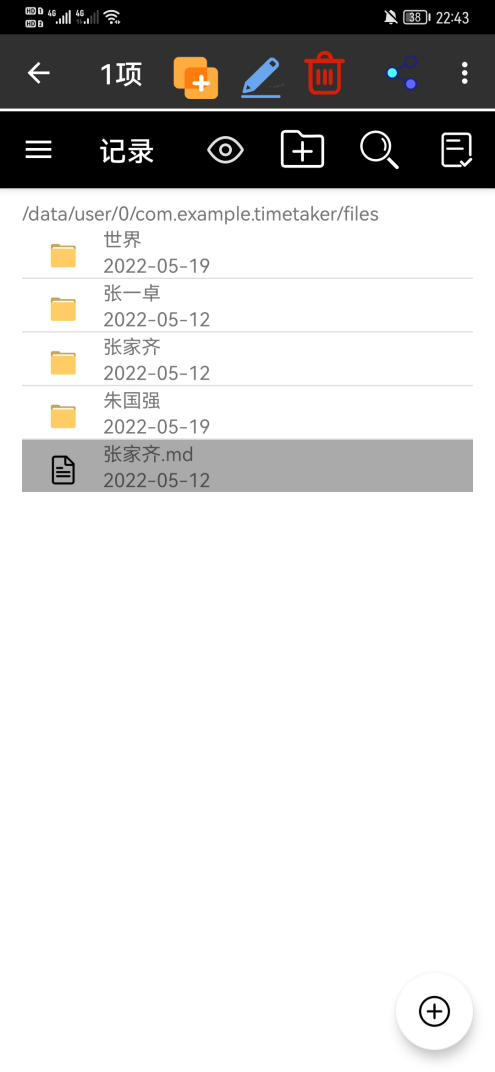
（3）Model层

Model层赋予了数据获取的职责，将数据的获取、存储、数据状态变化监听，获取网络数据接口，本地存储（增删改查）接口作为Model层的任务，提供数据获取接口供ViewModel调用。

**五、系统原型（设计）**

** **

（图5.1 侧边栏） （图5.2 Home界面）

** **

（图5.3 文件管理） （图5.4 编辑器）

（图5.5 日程管理） （图5.6 登录主界面）

**六、后期工作计划**

**1、后期工作**

(1)规范命名格式：对字符串、布局、踩点、Icon的命名进行统一修改。

(2)完善数据库：主要是在数据库中添加文件权限，同步相关属性。

(3)功能添加：待办事项与时间轴、云存储、语音输入、手写笔标注、图文混编、可拖动块结构......

(4)多人协同：登录状态下，实现多人协作共同完成一个文件记录。

**2、后期分工**

张一卓：Markdown渲染与待办事项、时间轴模块、回收站......

朱国强：文件查找、多人协同......

张家齐:数据库设计与云同步......

**3、日程安排**

|  |  |
| --- | --- |
| **日期** | **安排** |
| 5月15日前 | 规范命名格式 |
| 5月25日前 | 实现文件搜索、语音输入、代办事项与时间轴 |
| 5月30日前 | 实现云存储、回收站 |
| 6月10日前 | 实现手写笔标注 |
| 6月15日前 | 实现多人协同 |