1. **需求分析**

功能要求： 设计并实现一个个人日记本程序。

1）支持新增、修改和删除日记。日记为纯文本格式，日记的内容存储在SQLite数据库中，数据库表应该至少包含日记标题，日记内容，日记的创建时间，日记的作者等信息。

2）以ListView的形式显示当前用户的日记列表，点击某一条记录，可以打开一个新的界面供用户查看、编辑日记。

3）设计用户界面，维护日记的作者信息（至少包含作者的姓名），并存入SharedPreferences中。当应用程序启动时读取SharedPreferences中的作者信息，若不存在作者信息，则设置一个默认的作者姓名，用于新建日记的“作者”字段的值。

4）允许用户添加照片到日记中，方法是允许用户通过拍照或读取系统相册的方式添加照片，并将选中的照片拷贝到日记本应用程序的内部存储目录下，将照片文件名保存入数据库中与对应的日记记录关联在一起。用户查看日记时也显示日记中包含的照片。

非功能要求：界面按钮交互友好，意图清晰，使用户能够一目了然界面如何使用以及软件的功能。程序具有一定健壮性，要能够处理各种潜在的异常。

设计约束：界面约束，使用ConstraintLayout约束，以此来适配不同型号和屏幕大小的设备。

1. **软件设计**

说明：请在这里根据需求分析的内容，进行软件设计。需要阐明App采用的架构和实现方法，如所使用的主要安卓组件，功能模块划分，数据库设计等。

1. 用户一打开界面就知道如何使用，在User Information点击设置个人信息。一个显然的悬浮按钮“+”，点击创建新日志。
2. 利用RecylceView存放笔记内容的缩略，该类继承于ListView。数据库使用谷歌在2018发布的一个数据库操作工具：room，room能让开发者更好地操作SQLite。
3. **核心程序代码及运行结果**

本次显示日志列表使用RecycleView：

1. 继承重写RecyclerView.Adapter与RecyclerView.ViewHolder
2. 设置LayoutManager，以及layout的布局效果

区别：

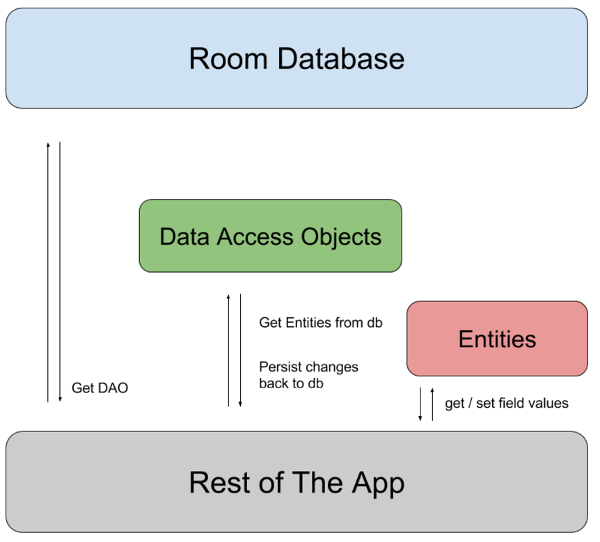
1. ViewHolder的编写规范化，ListView是需要自己定义的，而RecyclerView是规范好的；
2. RecyclerView复用item全部搞定，不需要像ListView那样setTag()与getTag()；
3. RecyclerView多了一些LayoutManager工作，但实现了布局效果多样化；

选择拍照和本地图片的时候需要申请权限：

@Override  
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults);  
 if(requestCode == *REQUEST\_CODE\_STORAGE\_PERMISSION* && grantResults.length > 0){  
 selectImage();  
 }else{  
 Toast.*makeText*(this, "Permission Denied!", Toast.*LENGTH\_SHORT*).show();  
 }  
}

private void selectImage(){  
 Intent intent = new Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*);  
 if(intent.resolveActivity(getPackageManager())!=null){  
 startActivityForResult(intent,*REQUEST\_CODE\_SELECT\_IMAGE*);  
 }  
}  
private void takeImage(){  
 file\_path = Environment.*getExternalStorageDirectory*() + File.*separator* +"Pictures"+File.*separator*+ new SimpleDateFormat("yyyyMMdd\_HHmmss").format(new Date())+".jpg";  
 Intent takeImageIntent = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
 if(takeImageIntent.resolveActivity(getPackageManager())!=null){  
 File imageFile = new File(file\_path);  
 try{  
 if (!imageFile.getParentFile().exists()) {  
 imageFile.getParentFile().mkdirs();  
 }  
 if (imageFile.exists()) {  
 imageFile.delete();  
 }  
 imageFile.createNewFile();  
 if(imageFile!=null){  
 mImageUri = FileProvider.*getUriForFile*(this,*FILE\_PROVIDER\_AUTHORITY*, imageFile);  
 takeImageIntent.putExtra(MediaStore.*EXTRA\_OUTPUT*,mImageUri);  
 startActivityForResult(takeImageIntent,*REQUEST\_CODE\_TAKE\_IMAGE*);  
 }}catch (Exception e){  
 Log.*e*("error",e.getMessage());  
 }  
 }  
}

数据库用ROOM，Room 在 SQLite 上提供了一个抽象层，以便在充分利用 SQLite 的强大功能的同时，能够流畅地访问数据库。处理大量结构化数据的应用可极大地受益于在本地保留这些数据。最常见的用例是缓存相关数据。这样，当设备无法访问网络时，用户仍可在离线状态下浏览相应内容。设备重新连接到网络后，用户发起的所有内容更改都会同步到服务器。

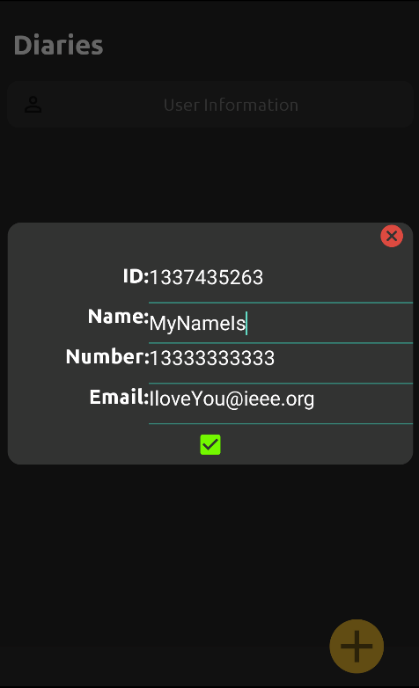
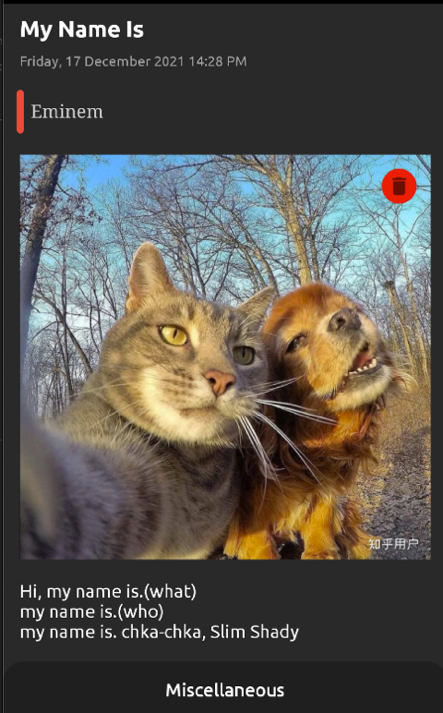
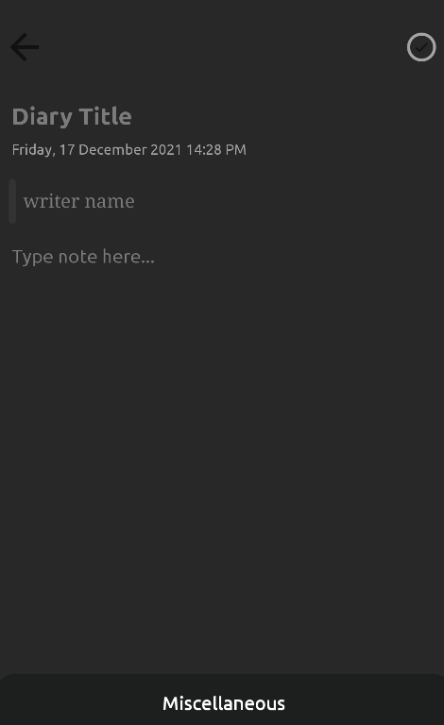
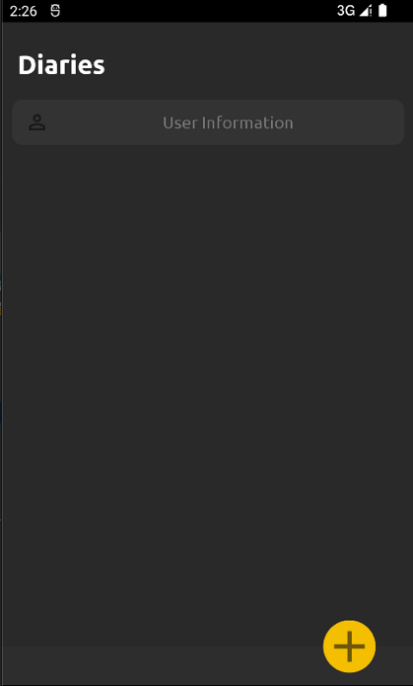
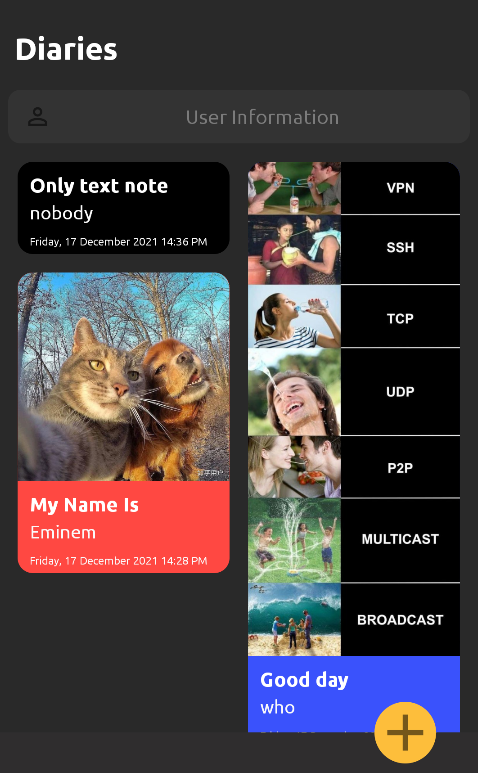
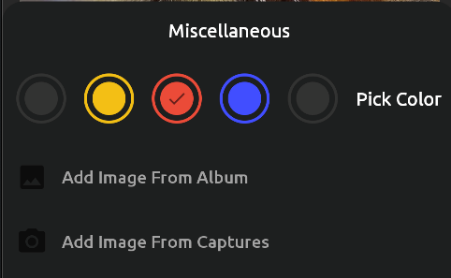


import androidx.room.ColumnInfo;  
import androidx.room.Entity;  
import androidx.room.PrimaryKey;

private String Color;

@Entity(tableName = "notes")

@PrimaryKey(autoGenerate = true)  
private int id;  
  
@ColumnInfo(name = "title")  
private String title;  
  
@ColumnInfo(name = "date\_time")  
private String dateTime;  
  
@ColumnInfo(name = "subTitle")  
private String subtitle;  
  
@ColumnInfo(name = "note\_text")  
private String noteText;  
  
@ColumnInfo(name = "image\_path")  
private String imagePath;  
  
@ColumnInfo(name = "color")  
private String Color;

执行结果如下：

**参考文献**

[1] 安卓官网文档Documentation for app developers <https://developer.android.com/docs>

[2] 课本《第一行代码》郭霖

[3] 使用 Room 将数据保存到本地数据库<https://developer.android.google.cn/training/data-storage/room?hl=zh-cn>

[4] Create dynamic lists with RecyclerView <https://developer.android.com/guide/topics/ui/layout/recyclerview>