# 一、概述

## 1.1项目名称

Sunny Weather

## 1.2项目产品的主要用途与性能

本项目为一个能显示给定城市天气情况和该城市天气预报信息软件，主要功能为根据用户提供的城市信息查询天气信息（包括实时天气情况、天气预报和生活建议）

## 1.3项目产品运行环境和测试环境

运行环境：操作系统版本为Android 5.0 以上的Android手机

测试环境：HUAWEI Nova 5 Pro

## 1.4功能需求分析

为用户提供一个简单的能显示天气情况及天气预报信息的软件，根据用户提供的城市信息查询天气情况及天气预报信息的服务。这个天气预报软件的主要功能应包括：

1.可以搜索全球大多数国家的各个城市数据；

2.可以查看全球绝大多数城市的天气信息；

3.可以自由切换城市，查看其它城市的天气；

4.可以手动刷新实时的天气。

## 1.5可行性分析

### 1.5.1技术可行性分析

软件基于Android Studio开发，架构为MVVM，主要用到一些UI、网络、数据储存、异步处理等技术，现有技术成熟，利用现有技术可以完成开发目标。

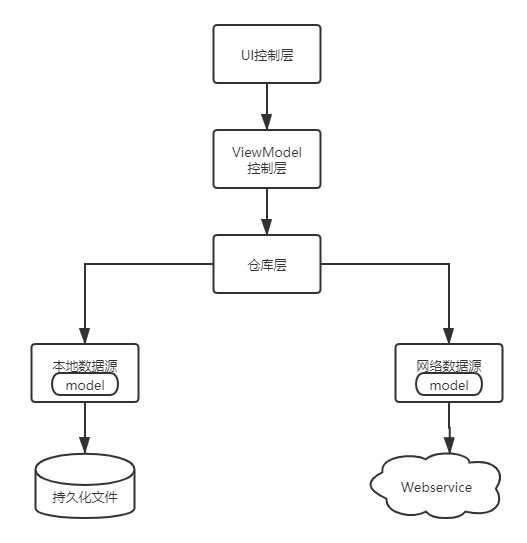
### 1.5.2资源可行性分析

主要是进行服务器接口选择，目前能查询到世界城市信息和对应的每个城市的天气信息的服务器接口越来越少，考虑到稳定性和安全性，我选择的是彩云天气APP的开放API接口。彩云天气的开放API不仅稳定且长期可用，还提供了全球一百多个国家的城市数据以及对应的每个城市的实时天气情况与天气预报信息，每天的申请上限是1万次每天。（注册网站https://dashboard.caiyunapp.com/）

# 二、项目架构设计

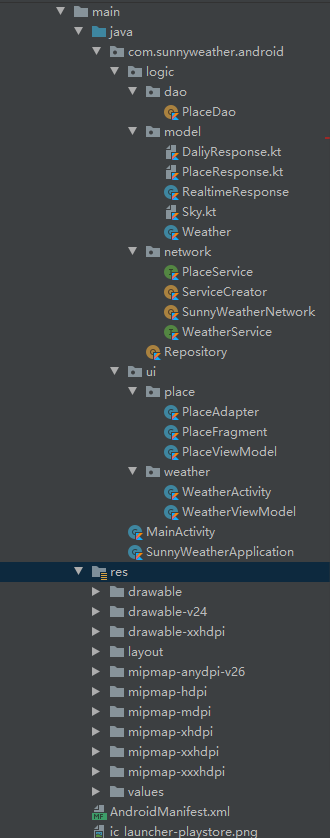
主要设计部分为MVVM架构，核心架构组件为Model、View、ViewModel，其中Model为数据模型部分，View是界面展示部分，ViewModel为连接Model和View的一个中转站，声明一些变量与具有基础功能的函数，从而实现让业务逻辑和界面展示分离的程序结构设计。除了这些核心架构组件之外，还应含有一些通用的模块组件，如仓库，数据源等，以上组件共同搭建了MVVM项目架构。

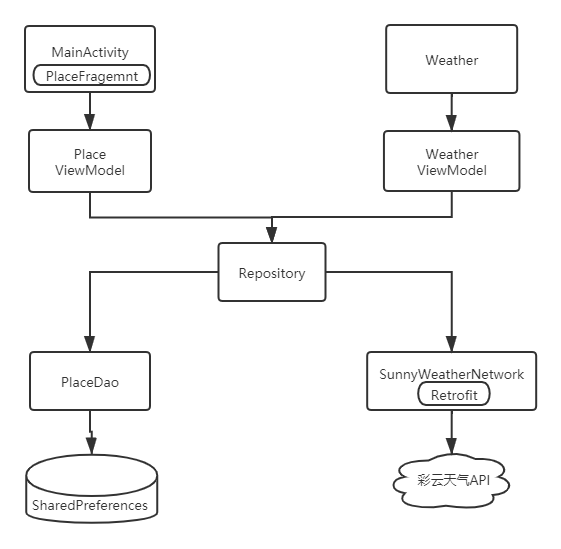
图1 MVVM项目架构图



由这个MVVM项目架构示意图可以很直观的得出一些信息，MVVM架构将程序分为若干层。UI控制层包括含有实现前端功能控制函数的活动文件（Activity）、布局文件（.xml文件），含有碎片控制功能函数文件（Fragment）等与界面相关的文件；ViewModel层用于持有和UI元素相关的数据，确保在屏幕旋转时数据不会丢失，且提供接口给UI控制层调用以及和仓库层进行通信；仓库层的主要工作是判断调用方请求的数据应该是从本地数据源获取还是网络数据源中获取，并将获取到的数据返回给调用方，其中，本地数据源使用数据库、SharedPreferences等持久化技术实现数据储存与调用，而网络数据源则通常使用Retrofit访问服务器提供的Webservice接口来实现。此外，项目架构图中单箭头表示的含义为箭尾所在层含有箭头所在层中的引用，反之则不成立，另外，引用也不能跨层持有。（代码层面上上是能实现的，但是有损代码的可读性和易维护性）

图2 项目具体架构、项目文件夹

本项目软件的核心功能为查询给定城市天气信息，要实现该核心功能，首先要解决的问题是实现搜索给定城市功能以及反馈给定城市的天气信息的结果页面，依次和MVVM项目架构对软件架构进行设计，具体设计图、项目文件夹结构如下。



可以看出，本项目所编写的代码是严格按照MVVM架构来实现的，且架构分层合理，在实现项目的过程中主要分为逻辑层实现和UI层实现两部分，在后续介绍功能设计实现过程同样分两部分描述。

# 三、详细设计

## 3.1搜索全球数据功能

### 3.1.1逻辑层实现

首先为了简化代码，设立全局变量方便调用，再根据对申请所得令牌值进行封装以方便调用。

根据端口返回JSON格式建立数据模型PlaceResponse以接收解析数据。（logic/model下）

建立Retrofit接口访问API数据：

1.建立PlaceService接口定义searchPlace()方法对服务器发出请求（logic/network下）

2.对所有网络请求的API进行封装，建立SunnyWeatherNetwork单例类（logic/network下）

建立仓库层，判断调用方请求的数据应从本地数据源中获取还是网络数据源中获取，并将获取数据返回给调用方；不过对于本项目来说，指定城市的天气情况应是实时变化的，因此建立本地数据库以储存天气信息必要性不高，主要是将向网络端口请求所得数据做好封装以方便调用。同时为了方便将异步获取的数据以响应式编程的方式通知给上一层，建立LiveData以实现此功能。

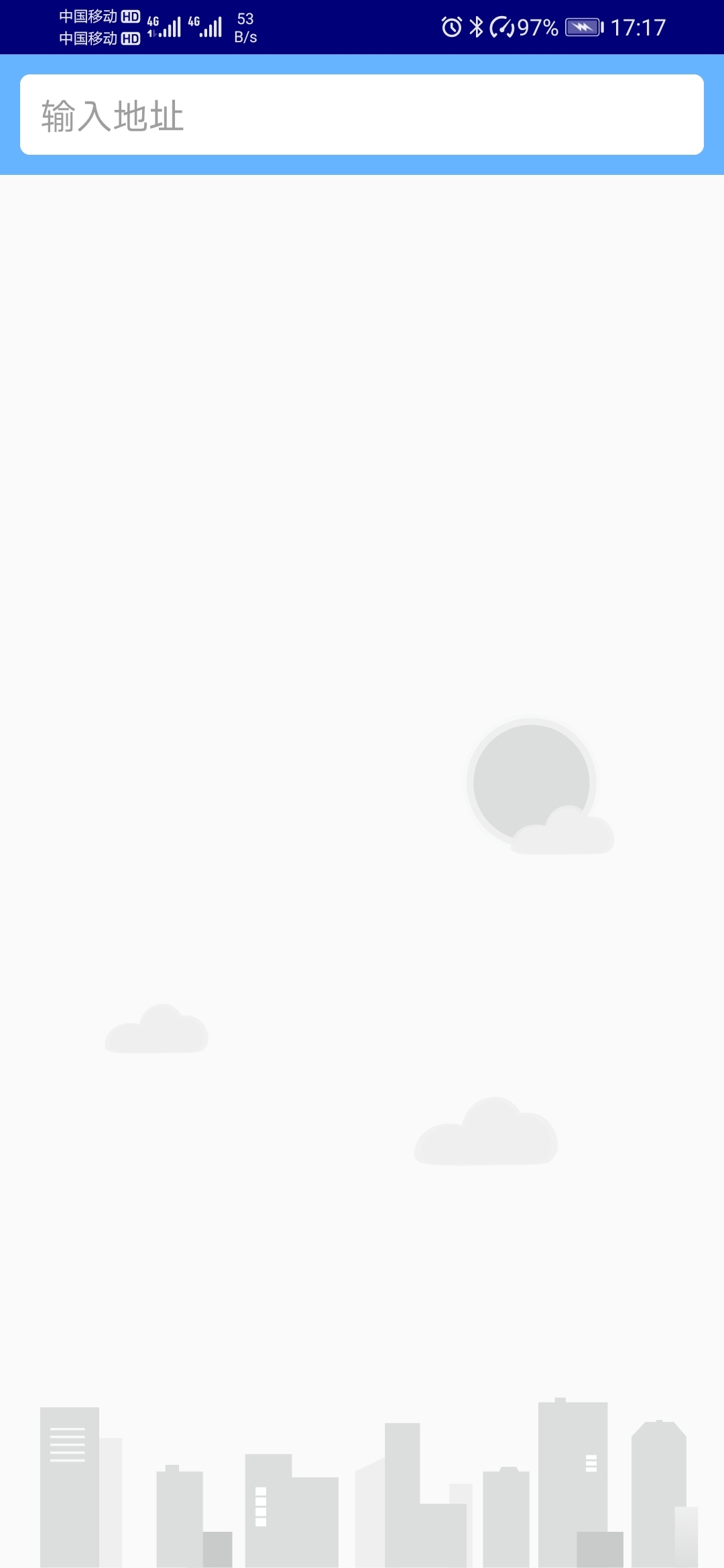
定义好数据源和仓库层后，接下来需要做的是定义ViewModel层，在项目架构设计中所提及的ViewModel层的主要功能为连接UI层和逻辑层的桥梁，同时对仓库层封装好的数据或者函数进行请求或者调用。建立PlaceViewModel(ui/place下)，其中定义searchPlaces函数，但函数并未直接调用仓库层的同名函数，而是通过调用LiveData实现的。此外，以方便用户，建立对象placeList以本地储存界面显示的城市数据。

自此，逻辑层的搭建工作基本结束了，接下来进行的工作为实现UI层代码

### 3.1.2 UI层代码实现

首先要实现布局文件，分析城市搜索功能主要的使用场景，首先是在启动页（初始化）应调用城市搜索功能页面，此外，在搜索结果页也应有按键或者手势控制弹出搜索页面以更换城市，因此，将搜索页处理成以fragment输出供需要时调用。另外分析城市搜索页面，搜索结果应以RecyclerView显示，子项的布局为卡片式布局，具体效果图如下：

图3 城市搜索结果页



其中搜索页主页面初始只含有一个输入栏和背景图，输入地址（城市名）时，根据所获得数据向网络接口（经过VM层、仓库层）发起请求，获得子项列表，这些子项布局为卡片式布局，而整体搜索结果是通过RecyclerView显示的。而RecyclerView与其他布局文件不同，不能单靠声明来实现调用，需要为其配置适配器，在ui/place下建立一个PlaceAdapter类（继承自RecyclerView.Adapter，且将泛型指定为PlaceAdapter.ViewHolder）。

以上工作完成后，在ui/place建立PlaceFragment文件与布局文件绑定，调用以及更改信息无需直接调用布局文件，调用PlaceFragment即可。

另外，在ui层代码实现刚开始就已经分析过，在用户第一次使用时（初始化），启动页面应是搜索页而非天气页面，因此，在主页面的布局文件中建立与fragment的联系。

## 3.2显示天气信息

### 3.2.1逻辑层代码

在实现搜索天气功能时已经搭建了网络环境，而在之前只是调用了Address信息，其他信息并未调用，在调用其他信息前要为其建造适配的模型以对应获取信息类型方便程序内调用，在logic/model包下新建RealtimeResponse文件对应从网络接口获得的实时天气JSON信息，新建DailyResponse文件对应从网络接口获得的未来几天天气JSON信息，然后，logic/model包下新建一个Weather类封装Realtime和Daily对象。

现定义一个用于访问天气信息API的Retrofit接口，在logic/network包下新建WeatherService接口以获取JSON数据。

然后再对仓库层文件进行修改，新增refreshWeather函数利用协程技术同时获取Realtime和Dailiy数据。

然后定义天气信息的ViewModel层，再ui/weather层下新建WeatherViewModel文件，声明同名函数供UI层调用，同时，为了保证屏幕翻转时数据不丢失，新建了几个变量储存页面信息。

### 3.2.2 UI层代码实现

首先要新建一个activity文件用于显示天气信息，再ui/weather包下新建WeatherActivity，布局文件名为activity\_weather。天气界面比搜索界面复杂多了，而且内容也更多，将不同的内容分开写在不同的布局文件中，再通过引入布局的方式集成到activity\_weather布局文件中，首先要做的是对天气结果页的布局信息进行分析：

首先，整体布局分为三大部分，分别用于显示当前天气信息天气预报信息和生活建议，接下来逐一分析各个部分UI界面与布局文件信息。

对于当前天气信息部分，分为两个部分，上半部分为标题栏，下半部分为当前天气信息，包括温度、天气和空气指数（JSON中不止这些数据）。

对于天气预报信息部分，主体为卡片式布局，内容为未来四天的天气信息，包括天气和温度信息。

对于生活指数模块，主体同样为卡片式布局，内容为四项具象化的生活建议，包括感冒指数（易发程度）、穿衣建议、紫外线指数、洗车指数。

另外，可以观察到整体布局较长，因此在最外部整体布局为Scroll，可以通过拖动显示屏幕外的信息。

图4 天气信息页面



UI界面编写完成后，在开始编写UI控制文件之前需要先将API获得的天气数据进行类型转换，在logic/model下建立Sky对象，将获得数据类型进行转换，主要输出为info、icon、bg三个字段，分别对应文字、图标、背景。

然后实现UI控制，在ui/weather下新建WeatherActivity，在其中请求天气信息，并将数据展示到页面上。

目前为止，本项目的基本功能部分已经完成了，接下来需要实现一些简单的用户逻辑。

## 3.3其他功能

### 3.3.1记录选中的城市

目前每次重新启动软件软件都会进入初始启动页即搜索页面，这是不符合用户逻辑的，因此软件需要自动记录当前选中的城市，显然需要用到本地储存功能，由于储存选中城市并非复杂数据，则只需使用SharedPreferences技术。

在logic/dao包下新建PlaceDao单例类，封装几个必要的储存和读取数据的接口，新建savePlace函数将Place对象储存在SharedPreferences中。跟储存对应的是读取，同样存在getPlace函数读取存储的数据，同时当数据存储时记录，以便启动软件时判断是否需要初始化。

### 3.3.2切换城市

目前虽然将搜索页设置为fragment以便调用，但是并未实现在结果页弹出搜索页，在结果也左上角添加一个按钮，设置按钮时间为弹出搜索页；另外，在布局文件中设置屏幕边缘左滑事件为弹出搜索页。

### 3.3.3下拉刷新

界面上的天气信息是可能过期的，并且并未设置自动刷新功能，需要为用户设计刷新方法，目前得到用户认可的刷新手段为下拉刷新。在布局文件外部嵌套一个刷新器，在天气信息页面添加一个函数即可实现下拉刷新。

# 四、测试用例

图5 初始启动页 图6 中文搜索城市 图7 英文搜索城市



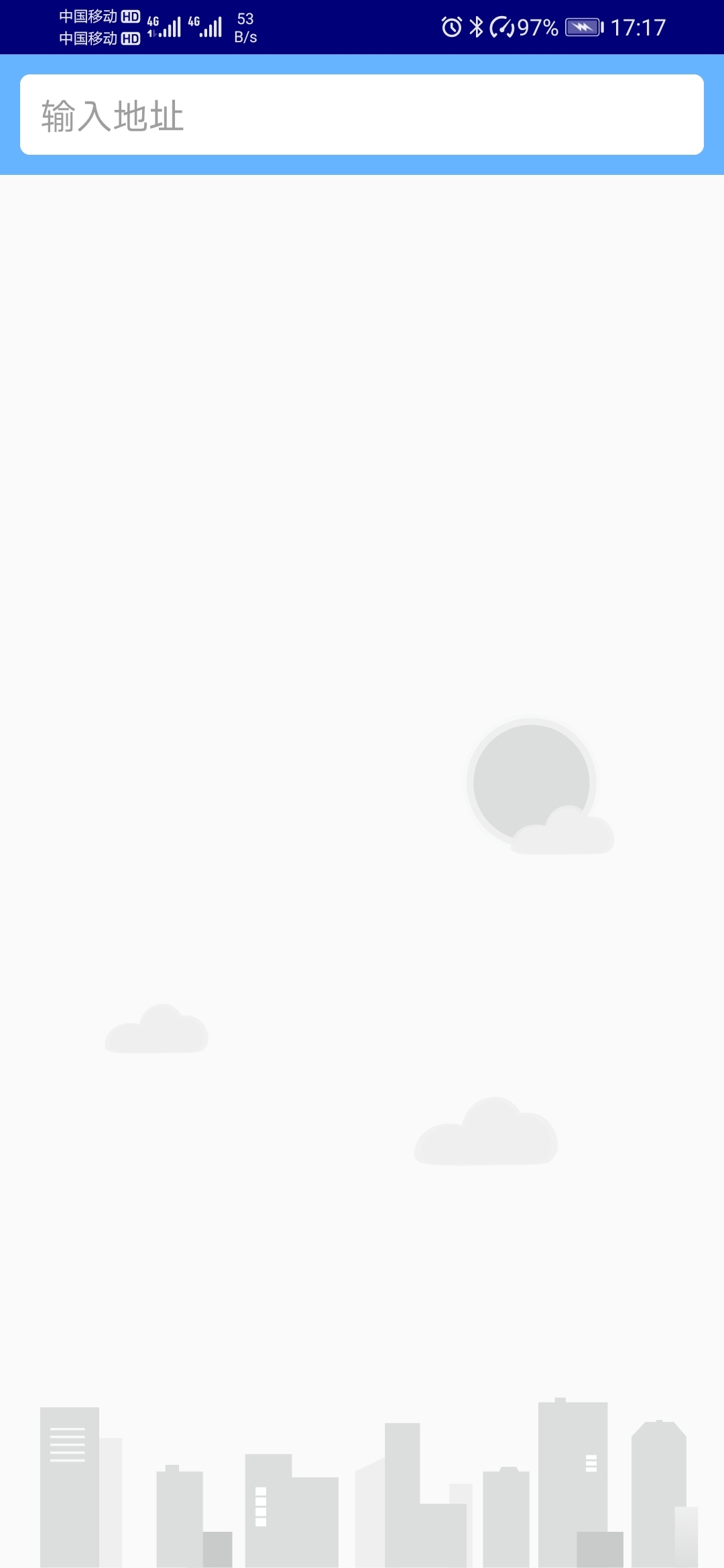


图8天气信息页 图9 切换城市 图10切换城市搜索页





图11 再次进入软件初始页



# 五、参考文献

[1]郭霖《第一行代码（第三版）》2020

# 六、附录

项目源码、日志：<https://github.com/enyushi/Weather.git>