实验三——实验报告

221220030 丁帅杰

一、根据 UML 图实现代码

IDE 选择 vscode，用 python 实现，ChatGPT-o1-mini 模型作为辅助。

（1）根据类图定义类

首先，详细分析 UML 类图，确定所有的类、属性、方法以及类之间的关系。

之后，根据类图中的继承，引用关系，确定类的层次结构，识别出哪些类是基类，哪些是派生类，哪些类之间有引用的关系，比如在 WeatherForeCast 类中有成员变量 WeatherData[]。

之后，分别创建各个类的文件（WeatherData.py,User.py 等），并添加成员变量和方法。其中方法暂时先不实现，等需要调用时再一一完善。比如在 WeatherForeCast 类中有变量 WeatherData[]。

```
class WeatherForecast:
    def __init__(self):
        self.weather_data_hour: List[WeatherData]=[]
        self.weather_data_day: List[WeatherData]=[]
```

（2）选用 python 库 tkinter 作为 UI 界面。

需要在 MainScreen 类中定义更多的变量和函数来显示各个界面的信息。

（3）根据时序图和活动图，完善各个类的成员函数。

时序图显示了用户和系统的行为和交互。根据时序图，一一实现相应操作所需的函数。

（4）调试和运行代码，发现并解决 bug

Python 是解释执行的，很多 bug 不会在静态 IDE 中显现。需要通过运行和测试发现 bug。

（4）界面美化

解决 bug 之后，尝试不同的字体和背景图，美化天气应用的 UI 界面。

最终实现的软件代码规模：

a. 总行数：约 800 行
b. 类数：共 9 个类
c. 方法数：约 30 个

二、大模型辅助代码实现

主要询问了 ChatGPT-o1-mini 以下几个问题：

（1）根据 UML 类图生成相应的类的 python 代码

在此步骤中，由于类中的方法暂时不用实现，因此这个工作并不需要多少思考，让大模型去完成可以节省时间。

大模型的结果：较完美地实现了各个类的代码，命名也符合规范。

（2）python 有什么 GUI 库？我要根据我的类图实现一个天气应用，推荐使用哪一个？

大模型的结果：推荐了 TKinter、PyQt 等 py 库，详细地阐释了不同库的优点和缺点。由于实现的天气 App 还是比较轻量级的程序，我选择用轻量级的 TKinter。

（3）Tkinter 怎么使用，有哪些函数？举几个例子。

大模型结果：从引入 tkinter 库开始，较为详细地列举了 Tkinter 的常用函数，比如：

```
root = tk.Tk()
root.mainloop()
tk.canvas.create_text()
tk.Button()
tk.Entry()
```

并生成一段可执行的代码样例，让我可以直观的看到运行结果。

在之后的具体实现中，根据需求向大模型询问了更多 tkinter 的函数，比如：鼠标事件绑定、如何显示图片、Tkinter 可以使用哪些字体等。

（4）Tkinter 显示的按钮会有带颜色的背景，怎么消除（只留下文本）？

大模型的结果：给出一段定义 Button 的代码，运行后发现，背景并未消除，大模型回答不正确。

我的解决方法：用 create_text()函数创建文本，代替按钮。并在文本区域绑定鼠标事件：鼠标左击文本区域，触发相应函数。但为了代码简洁，保留了一部分按钮。

（5）实现搜索城市的功能，怎么获取一个全球城市的列表？

大模型的结果：推荐从 GeoNames 下载 cities1000.zip 并解压，获取 cities.txt 文件。该文件包含了全球超过 1000 个以上人口的城市。

（6）python 有什么 API 可以让我获取某个地区的实时天气信息？

大模型的结果：推荐了几个平台：OpenWeatherAPI，WeatherAPI.com，Open-Meteo 等，并生成了正确的示例代码解释如何使用 API。

总结：大模型在辅助开发的过程中提供了有效且重要的帮助，节省了很多查询文档的时间。但不能过度依赖大模型，也不能完全信赖大模型给出的代码。

三、代码编译和运行

代码已实现实验一中需求分析的绝大部分功能，并和实验二中 UI 界面一致。

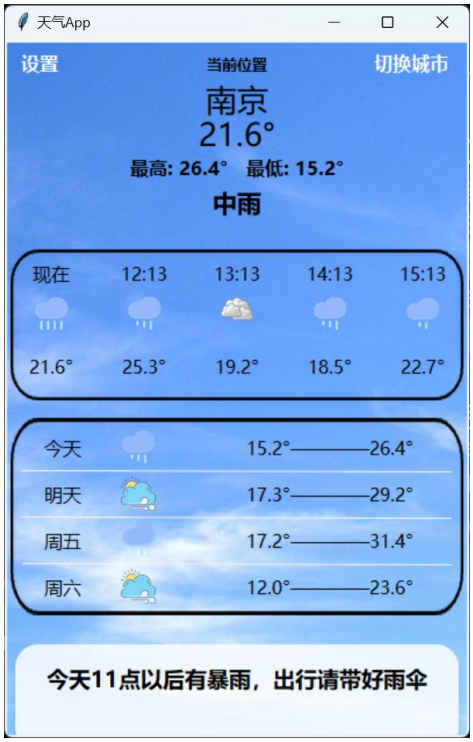
在 code 文件夹下，city 文件夹中包含了城市列表，pic 文件夹为 UI 界面所需的图片，weather_app 中是各个类的 py 代码。

运行 main.py，启动程序，以下是各个功能的运行结果和截图：

（1）登录页面：

第一次登录时，只有一个用户信息（ID : root , password : 123）。用户可以输入这个账号密码登录。也可以选择注册，注册后用户信息将保留在本地数据库中。

（2）登录后，显示天气应用的主界面，包含当前城市、实时温度、小时级天气预报、几天内天气预报、提醒信息等。



（3）点击小时级预报的黑框，可以查看当日具体信息：

包含小时的天气预报、空气质量、风力和风速、能见度、空气湿度、降雨概率等信息。

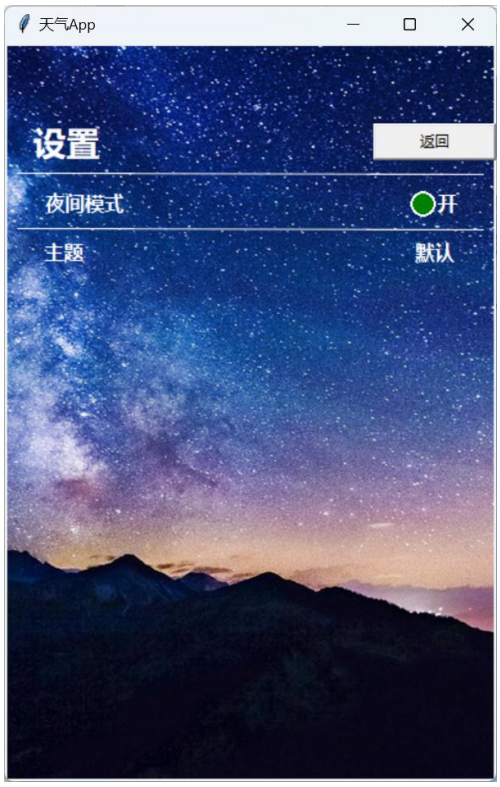
（4）返回主界面，点击两周内天气预报的黑框，可以查看两周内的天气预报：

（5）返回主界面，点击切换城市：

显示了收藏中的城市。可以直接切换至收藏的城市，也可以搜索城市，将搜索到的城市加入收藏或直接切换。

（6）点击主界面左上角的“设置”，进入设置界面：

在设置中可以开/关夜间模式



有待实现的功能：

（1）**主题：**目前主题只有默认主题。后续可以添加更多主题，不同主题有不同的图标，使应用更个性化。

（2）**实时天气：**对于 OpenWeatherAPI 等天气 API，免费服务无法满足需求（14 天、24 小时的天气预报），需要付费服务。因此暂时显示的天气数据都是本地生成的，不是实时的数据。后续加入该功能时，只需更新 weatherForecast 类中获取天气的函数即可。

（3）**城市搜索功能：**cities1000.txt 文件数据过多，引入 tkinter 时程序卡顿崩溃。因此我提取了城市人口大于 100000 的中国城市，并未实现全球范围内任意城市的搜索。有待后续实现。

（4）**界面：**实验二用 Pixso 设计的 UI 界面，其中的字体无法获取，本应用中的字体为“微软雅黑”。因此代码实现的 UI 界面没有与 Pixso 设计的界面完全一致。

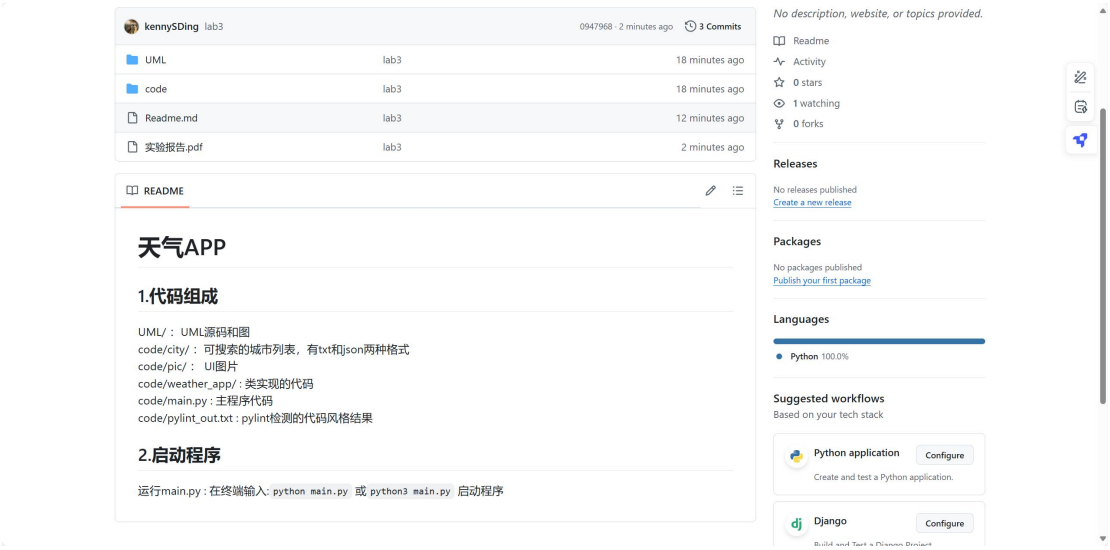
三、代码风格

用 pylint 检测了代码风格，检测结果在 pylint_out.txt 文件中。

四、Git 远程代码管理

已创建 Github 远程仓库，并将本地文件推送至远程仓库，并编写了 Readme.md。

远程仓库链接：https://github.com/kennySDing/NJU-SF-lab


kennySDing lab3
0947968 · 2 minutes ago
3 Commits
UML lab3 18 minutes ago
code lab3 18 minutes ago
Readme.md lab3 12 minutes ago
实验报告.pdf lab3 2 minutes ago
README
天气APP
1.代码组成
UML/ ： UML源码和图
code/city/ ： 可搜索的城市列表，有txt和json两种格式
code/pic/ ： UI图片
code/weather_app/ : 类实现的代码
code/main.py : 主程序代码
code/pylint_out.txt : pylint检测的代码风格结果
2.启动程序
运行main.py : 在终端输入: python main.py 或 python3 main.py 启动程序
No description, website, or topics provided.
Readme
Activity
0 stars
1 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
Python 100.0%
Suggested workflows
Based on your tech stack
Python application
Configure
Create and test a Python application.
Django
Configure