

图一：UML类图

在本项目系统中，我们运用了GOF设计模式对大作业进行了详细设计，下面列举其中两种设计模式在本系统中的应用。

一：我们在系统中运用了Decorator模式：

使用Decorator模式，动态的为User增加了管理用户和管理景区景点的功能，生成了Administrator（系统管理员）和Scenic Operator（景区管理员）两个对象。装饰模式以对客户端透明的方式扩展对象的功能，是继承关系的一个替代方案。就增加功能来说，Decorator模式相比生成子类更为灵活。不改变接口的前提下，增强所考虑的类的性能。装饰模式可以提供比继承更多的灵活性。装饰模式允许系统动态决定“贴上”一个需要的“装饰”，或者除掉一个不需要的“装饰”。继承关系则不同，继承关系是静态的，它在系统运行前就决定了。所以此时使用Decorator模式更为妥当，便于更好的管理用户的角色。

二：我们在系统中运用了Template模式：

使用Template模式，实现了一个自动生成Travel Note的模板，根据用户旅游的足迹和评论，生成属于用户本身的Travel Note。使用Template模式可以定义一个操作中算法的骨架，而将一些步骤延迟到子类中，模板方法使得子类可以不改变算法的结构即可重定义该算法的某些特定步骤。同时也存在一种反向的控制结构，通过一个父类调用其子类的操作，通过子类对父类进行扩展增加新的行为，符合“开闭原则”。在我们的旅行日记生成中，首先定义一个Travel Note的模板可以使得代码复用，同时根据用户的自主设置，或者后续根据用户对自动生成的游记的修改进行学习，从而获得更加贴合用户体验的旅行日记。