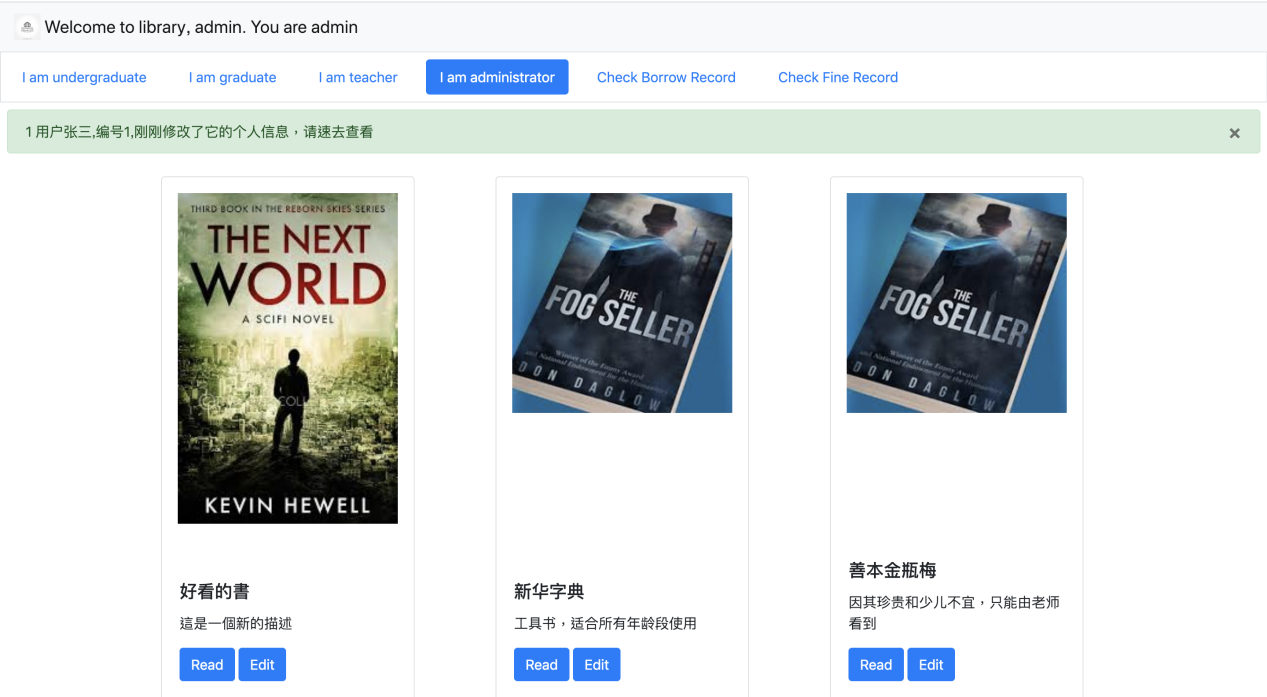
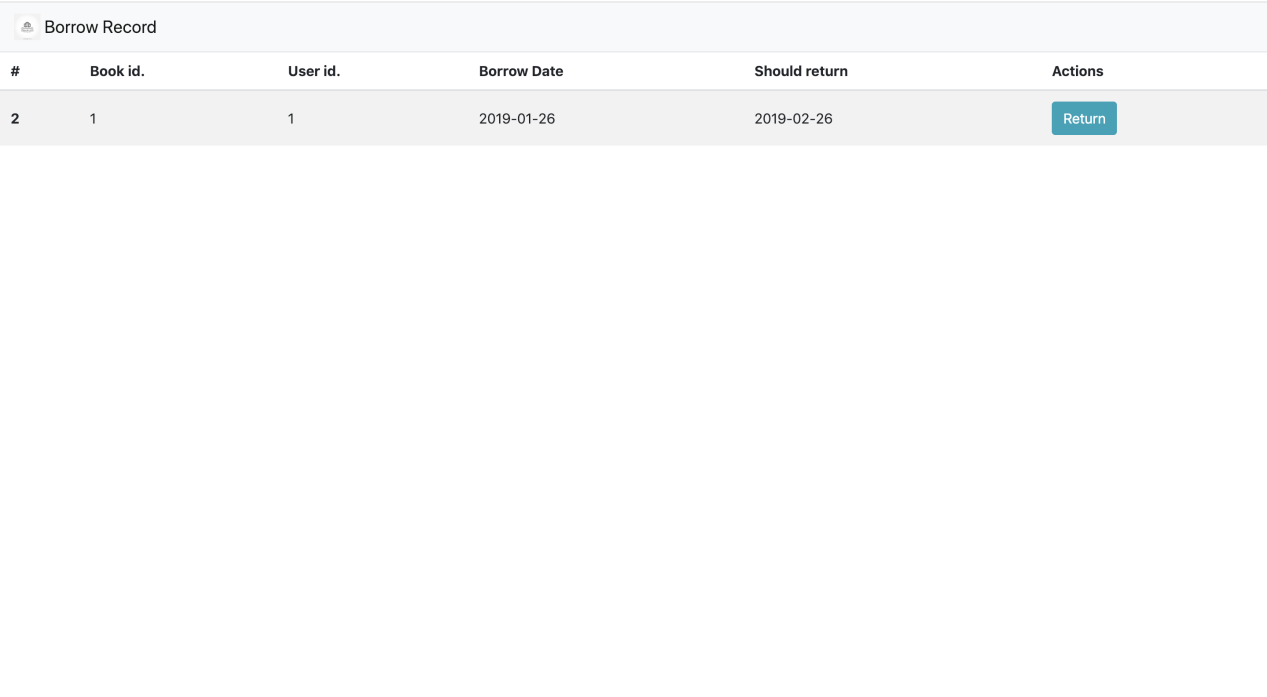
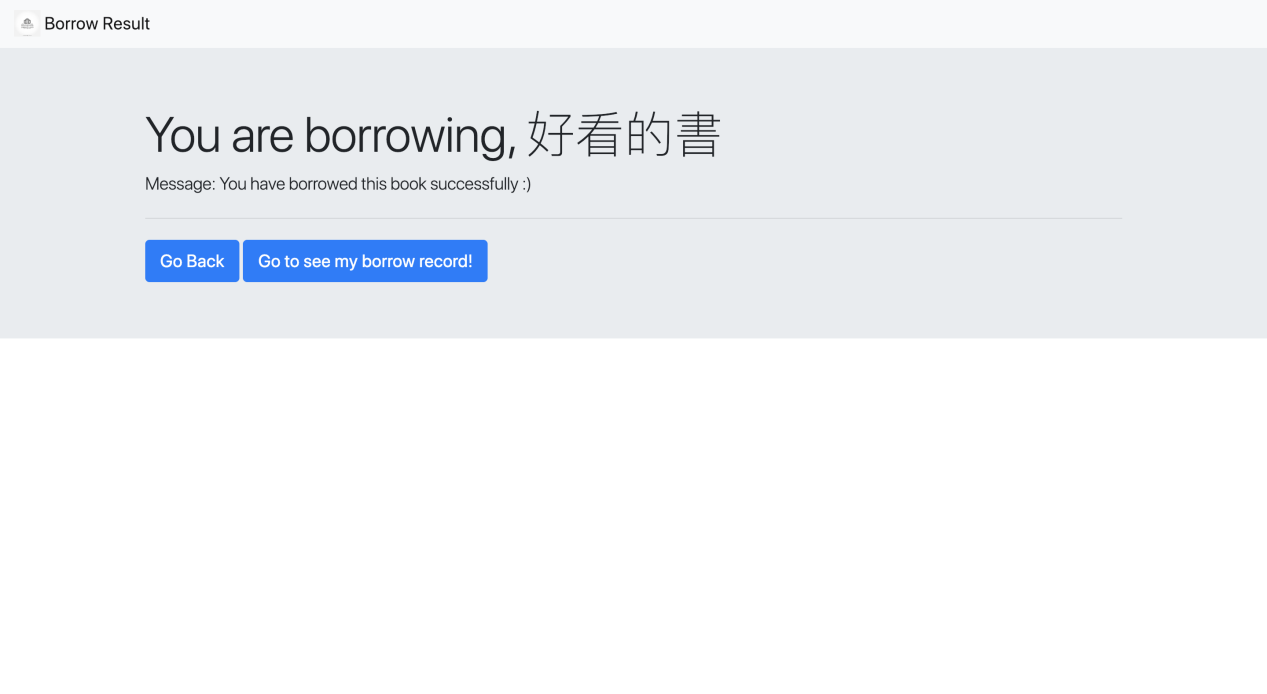
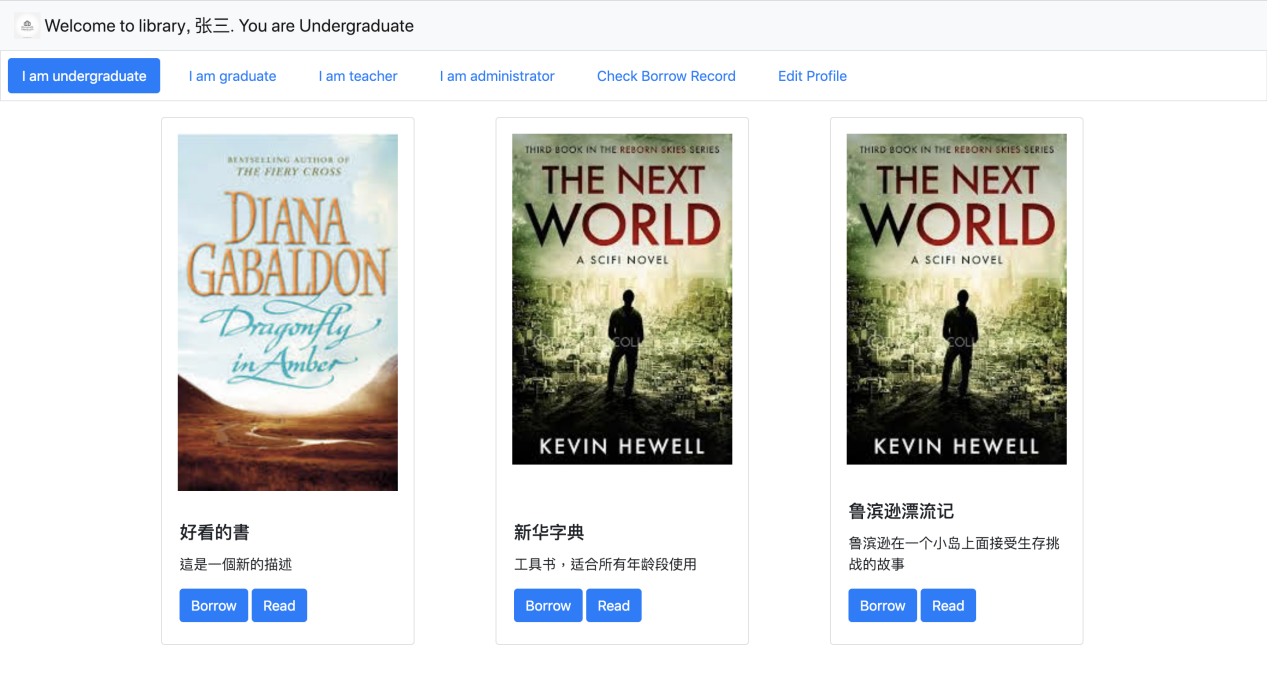
軟件系統設計和體系結構 作業三 組員： 161250047 吉宇哲 161250051 賴健明

介面截圖

1. 管理員主頁

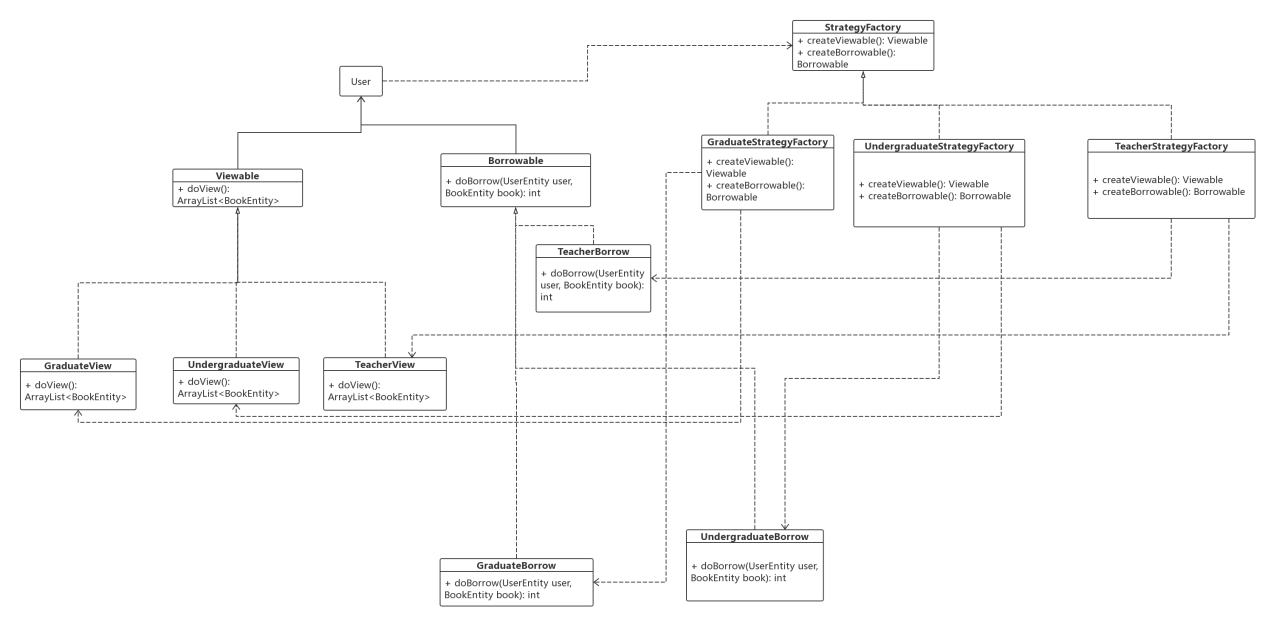
2. 借書人查看借書紀錄3. 借書結果

4. 本科生主頁

對第一部份的設計

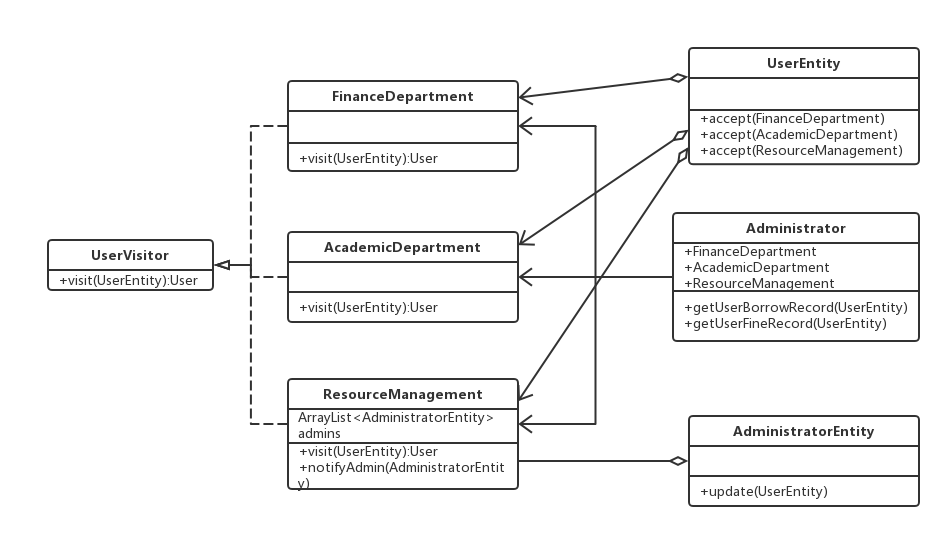
第一部份要求能夠對於本科生、研究生和老師使用不同的借書行為和查看權限內的圖書，而且需要支持未來新增加的角色的借書行為，故該模塊需要較高的可擴展性，加上變化的部份是對於某個行為的具體實現，因此採用了策略模式。為抽離創建具體策略類的複雜度，和配合借書和查看圖書兩個產品族，因此對於具體策略類的創建採用抽像工廠類。

User類持有抽像策略類Borrowable和Viewable，且在初始化會使用抽像工廠來創建出具體策略類，包括借書的具體策略和查看圖書的具體策略。借書的具體策略類包括GraduateBorrow，TeacherBorrow，UndergraduateBorrow。查看圖片的具體策略類包括GraduateView，TeacherView，UndergraduateBorrow。對於增加的新角色，可通過增加新的View和Borrow兩個產品族的具體策略則可實現，但對於增加新的產品族支持則較差。



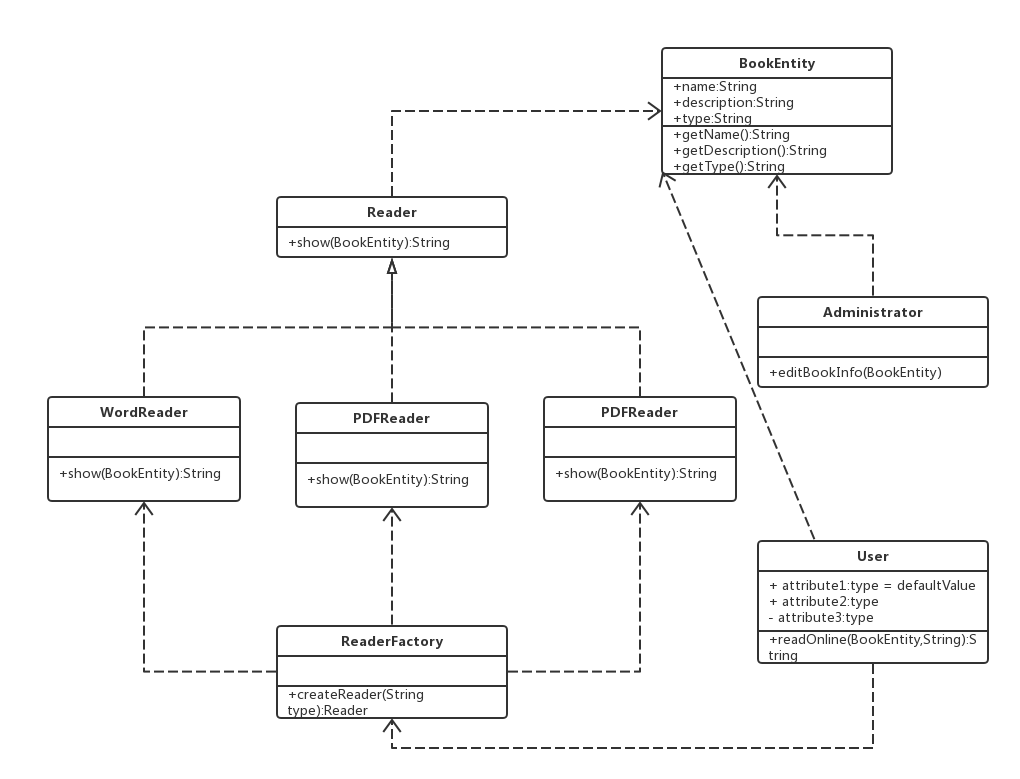
对第二部分的设计：

第二部分要求管理员能够管理用户的信息，例如生成用户的借书记录和生成罚款记录，且在未来有很大的可能性会有更多的管理功能，并且要求管理员能够将一些管理功能授权给学校的其他部门或者院系。所以这对设计的可拓展性要求比较高，用户的属性且不同的操作访问的是用户的不同数据，在这里用户的属性变化的可能性不高，但是针对用户的不同操作的变化和增加可能性较高。所以我在这里考虑使用了访问者模式，将对用户不同属性进行访问和操作的方法分发给不同的访问者类。当前FinanceDepartment拥有生成罚款记录的功能，AcademicDepartment拥有生成结束记录的功能，ResourceManagement拥有修改用户属性的功能。这样如果增加新的操作，就增加一个新的访问者类即可。将对用户的操作和用户的属性解耦，并且将不同应用场景的代码解耦，不在彼此依赖，这样可以尽量避免每增加一个新的操作都要修改用户类的代码。并且这样做的另一个好处是，将操作分发给不同的对象，这样每个访问者的权限都相对较小，然后通过管理员去关联这些访问者类，从而实现管理员的所有功能，当任何一个部门想要拥有对用户的部分访问权限的时候也可以通过组合不同的访问者访问用户的部分属性，而不是将所有权限授权给那个部门。这样运用组合更加符合面向对象的思想和要求，并且这样只有一个方法的“瘦接口”符合接口隔离原则。



对第三部分的设计：

第三部分要求用户可以能够在线通过不同版本的阅读器阅读书籍资源，比如Word、PDF和EPUB，我认为阅读器是存在拓展的可能性的，所以让所有阅读器继承Reader抽象类并且实现show方法，在这部分中我使用了简单工厂模式来通过用户传进来的参数创建不同的阅读器，这样的设计更符合用户的直觉，将具体阅读器的创建和用户解耦，在更换或者添加阅读器种类和具体实现的时候不用修改用户代码。虽然可以采用工厂模式来创建不同工厂来生成不同的阅读器，但是我觉得那样设计会显著增加类的个数和系统的复杂性，有一些“杀鸡用牛刀”。最后就采用了简单工厂模式。



对第四部分的设计：

第四部分要求用户能够修改用户的资料和个人信息，并且在用户修改个人信息后所有管理员能够收到自动生成的邮件作为提醒，这是很明显的观察者模式的运用。接着第二部分的设计我们清楚ResourceManagement拥有修改用户信息的功能，只需要用户类持有ResourceManagement类的引用即能让用户可以修改个人信息，ResourceManagement持有所有登记在册的管理员的列表，当调用修改功能的时候对列表中u搜友管理员调用管理员的update方法对管理员进行通知，update方法中生成一个邮件保存到数据库中就实现了对管理员的邮件提醒功能。

重要的功能需求列表

|  |  |
| --- | --- |
| **接口名** | **需求内容** |
| **User.borrowBook** | 实现用户的借书功能，根据不同的用户自动调用不同的借书方法 |
| **User.getBookList** | 实现用户的查看图书列表功能，根据不同的用户自动调用不同的方法获取他们所能看到的图书列表 |
| **User.returnBook** | 实现用户的还书方法，判断用户是否超期，超期自动生成一条罚款记录且自动计算应罚金额 |
| **User.readOnline** | 实现用户的在线阅读方式，通过传入的类型自动生成不同阅读器进行阅读 |
| **User.updateUserInfo** | 实现用户的更新信息功能，更新信息后自动发送邮件给管理员进行通知 |
| **Administrator.EditBookInfo** | 管理员修改图书的信息 |
| **Administrator.getUserBorrowRecord** | 管理员获得用户的借书信息 |
| **Administrator.getUserFineRecord** | 管理员获得用户的罚款信息 |
| **Administrator.getBookList** | 管理员获得所有图书馆的图书以便进行信息修改 |

個人貢獻和收獲

吉宇哲：

我们这次的分工是每个人负责两个Component的设计和实现，我设计的是第二部分和第三部分，数据库的表结构是两个人一起商量共同完成的，通过这一次对这两个Component的设计和从后端到前端的实现，我巩固和锻炼了前后端的开发能力，对设计的反复斟酌和挑选合适的设计模式让我对常用的设计模式有了更深刻的了解，对各种设计模式的优缺点也有了更加深刻的体会，对他们所体现出来的设计思想也有了更加深刻的了解。日后在设计其他项目的时候我会首先考虑能否运用设计模式来帮助设计出高容错，低耦合，强拓展性的项目。

賴健明：

我負責第一個組件和第四個組件和設計和實現，通過這次作業，再次加強了我的設計和開發能力，對設計模式有進一步的了解和掌握，尤其對於策略模式和抽像工廠模式，對於它們的使用場景和優點缺點也有更了的認知，在實際編碼時，體會到使用合適的設計模式對開發效率的提升，也提高了組員之間的溝通效率。