### **一、设计与实现**

**（1）设计思想**

目的是设计一款基于Python开发语言的图书管理后台系统，为业务后台提供一个图书管理的平台，便于业务人员进行对图书的管理。代码架构是基于MVC（模型Model-视图View-控制器Controller）的设计模式思想，并且也是用到了DTO思想，对于Model添加了需要用到了的DTO，减小了模块间的耦合，加强了模块内部的内聚，使其更具有结构性。

**（2）总体设计**

根据需求可以分为两个界面：登录界面和图书界面

登录界面的设计：以防直接进入图书管理界面，只有管理员帐号可进，而管理员账号密码，可在后端代码对应处自行更改。此作用仅仅是一层简单防范，也可进行扩展。

图书界面：基于分页展示所有图书的信息，具有增删改查功能，可以对自己感兴趣的书籍区域部分进行搜索查询，可以添加新的书籍信息，可以修改已添加的书籍信息，还可以删除不想要的书籍信息。

综上所述，前端编写两个界面以及界面的组件调用实现，后端编写登录的接口以及能连上数据库并能对图书进行增删改查的接口。

**（3）模块设计**

登录功能：前端点击登录按钮后，调用后端登录接口，后端进行账号密码参数的校验，不能为空，进而进行判断是否是正确的管理员账号密码，从而返回结果。

查找功能：前端点击查询按钮后，调用后端List查询接口，后端根据传入的参数从所有图书信息中，进行筛选满足条件的数据。其中对于书名和作者两个参数，如果传过来的为空，则不对这两个字段进行筛选，否则进行模糊匹配查找并筛选；对于书籍类型参数，如果传过来的为0，则表示全部类型，那么也不对此字段进行筛选，否则筛选出传入过来的类型的书籍信息数据。

增加功能：前端点击增加按钮后，在弹出的弹框中将对应书籍信息填充后，点击确认增加按钮，前端对参数进行校验，有一项为空则会弹出不能为空的提示。校验完后，调用后端增加接口，在数据库里插入这条数据，然后前端刷新后则会显示出来刚刚增加的书籍信息。

修改功能：前端在某一条书籍信息后点击此修改按钮，在弹出的弹框中将对应书籍信息修改后，点击确认修改按钮，前端对参数进行校验，有一项为空则会弹出不能为空的提示。校验完后，调用后端修改接口，后端根据传过来的Id参数获取原来的书籍信息，然后将该信息对应字段进行修改，然后在数据库里保存修改的这条数据。前端刷新后，能查看到刚刚修改的书籍信息已更改。

删除功能：前端在某一条书籍信息后点击此删除按钮，在弹出的弹框中可以选择确认修改，点击后则会调用后端修改接口，后端通过传过来的Id参数获取书籍信息，并将IsDeleted字段标记为true，进行软删除。在数据库中保存修改后，前端刷新页面后，就查看不到刚刚对应的书籍信息。若在弹框中选择取消修改，则弹框消失，返回原页面。

**（4）数据库设计**

使用的数据库是MySQL，数据库管理工具是DBeaver,，创建了一张图书表。用MySQL创建此图书表语句如下：

CREATE TABLE `book` (

`Id` int NOT NULL AUTO\_INCREMENT COMMENT '主键Id',

`BookName` varchar(20) NOT NULL COMMENT '书名',

`Author` varchar(20) NOT NULL COMMENT '作者',

`Type` int NOT NULL COMMENT '书籍类型',

`Description` varchar(200) NOT NULL COMMENT '简介描述',

`CreateTime` datetime NOT NULL COMMENT '创建时间',

`IsDeleted` bool NOT NULL COMMENT '是否删除',

PRIMARY KEY (`Id`)

)

**（5）开发环境**

VSCode。可以安装一些需要用到的扩展，比如前端的Live Server扩展，以便进行浏览器页面展示；后端的Python扩展，以便进行后端Python代码的编写。

**（6）相关技术**

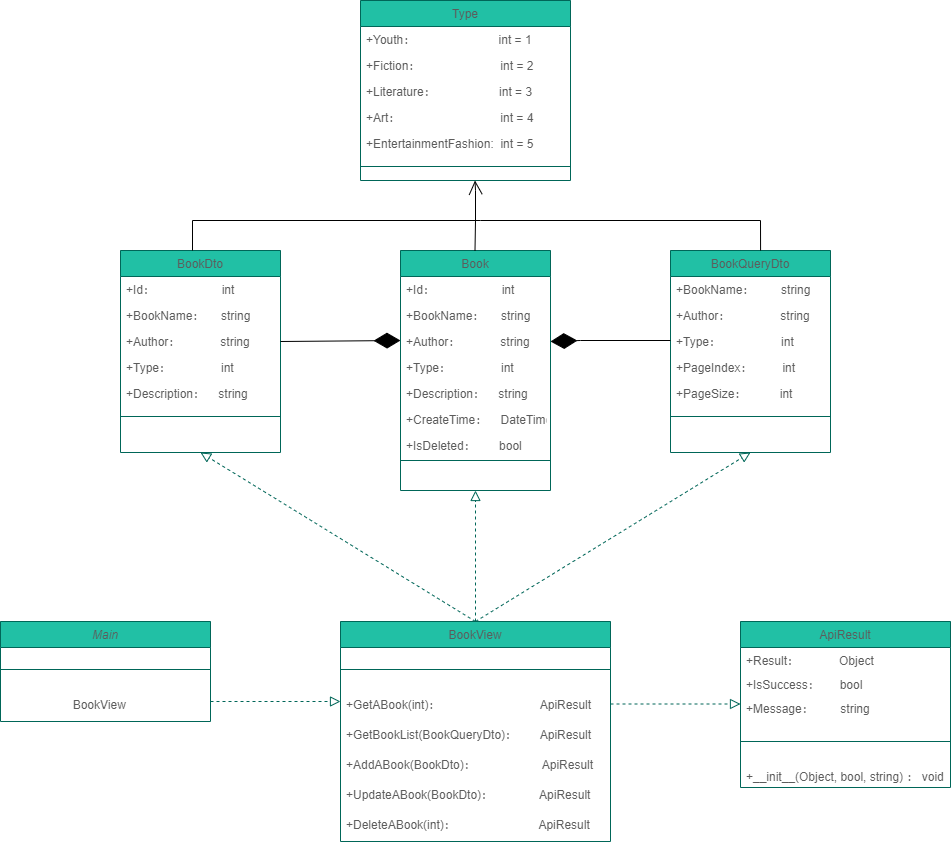
后端：引入FastAPI包，使用Python的FastAPI框架进行后端接口的代码编写。引入SQL alchemy包，从而对数据库进行连接并且能进行数据库对应表的增删改查操作。使用FASTAPI提供的的中间件进行后端的跨域配置，以免前端访问后端接口时出现跨域的相关报错。引入uvicorn包进行后端的api环境的运行。开发期间可以使用此框架提供的swagger页面进行接口的测试。

前端：使用HTML、JS、CSS进行前端界面的布局、样式和执行操作的编写，同时尽量去美化了页面中相关组件的样式。引入了JQuery包，便于JS代码的编写。

引入了BootStrap包，便于按钮样式的选择。

### **二、UML类图相关**

**（1）UML类图**



**（2）对类属性的解释**

|  |  |
| --- | --- |
| Youth | 青春类型 |
| Fiction | 小说类型 |
| Literature | 文学类型 |
| Art | 艺术类型 |
| EntertainmentFashion | 娱乐时尚类型 |
| Id | 主键Id |
| BookName | 书名 |
| Author | 作者 |
| Type | 书籍类型 |
| Description | 书籍简介 |
| CreateTime | 创建时间 |
| IsDeleted | 是否删除（用于软删除） |
| PageIndex | 当前页 |
| PageSize | 一页大小 |
| Result | 接口返回值 |
| IsSuccess | 接口是否调用成功 |
| Message | 调用接口失败原因 |

**（3）对接口函数的解释**

|  |  |
| --- | --- |
| GetABook(id: int): | 根据入参Id获取对应书籍信息 |
| GetBookList(dto: BookQueryDto)： | 根据入参的书名、作者进行模糊查询，以及根据书籍类型、当前页和每页大小进行筛选数据，进而得到分页的查询数据 |
| AddABook(dto: BookDto)： | 根据入参信息进行增加书籍 |
| UpdateABook(dto: BookDto)： | 根据入参的Id获取对应书籍的原信息，再根据剩下的入参对其进行更改 |
| DeleteABook(id: int)： | 根据入参Id找到对应书籍，并对其进行删除 |

### 三、使用帮助

1. 安装Python需要使用到的包，包括Python使用到的框架包、启动包和数据库连接及操作包。最好在管理员权限下执行以下安包操作：

pip install fastapi

pip install uvicorn

pip install mysql

pip install pymysql

pip install sqlalchemy

（2）在后端setting.py文件中，可以更改数据连接的配置，主要是更改账号名、密码、数据库名。注意：该文件夹下的HOST和PORT不要更改，否则前端调用接口的路由会与后端不一致，导致调用接口时出错。

（3）在此项目文件夹下，执行python .\main.py 启动后端。

（4）打开http://127.0.0.1:8000/docs 能看到后端api的swagger，供开发调试接口使用。

（5）运行完后端后，运行前端文件夹static/html路径下的index.html文件。可以使用上文说的Live Server扩展，右击该文件，选择Open With Live Server即可展示此文件的前端页面。

（6）在登录页面下，输入后端定义好的的管理员帐号密码后即可进入图书管理系统界面。账号密码可以在view/user.py代码中看到，也可以自行更改。