

**Bookstore——网上购书功能网站后端**

|  |  |  |  |
| --- | --- | --- | --- |
| **课程** | **当代数据管理系统** | | |
| **姓名** | **唐小卉** | **刘钊瑄** | **仲韦萱** |
| **学号** | **10215501437** |  |  |

**2023年 11 月**

目录

[Bookstore——网上购书功能网站后端 1](#_Toc7843)

[1、 实验要求 3](#_Toc18361)

[2、 项目运行 4](#_Toc17746)

[3、 数据库设计 4](#_Toc10919)

[4、 功能实现 4](#_Toc17974)

[4.1 用户功能 4](#_Toc7953)

[4.1.1 token与jwt 4](#_Toc10077)

[4.1.2 register登录 5](#_Toc21909)

[4.1.3 check password检查密码 5](#_Toc13961)

[4.1.4 logout登出 6](#_Toc19971)

[4.1.5 unregister注销 6](#_Toc8448)

[4.1.6 change password更改密码 6](#_Toc17019)

[4.2 卖家功能 7](#_Toc6820)

[4.3 买家功能 7](#_Toc30862)

[4.4 发货收货 7](#_Toc5907)

[4.5 搜索图书 7](#_Toc4199)

[4.6 订单状态 7](#_Toc27820)

[5、 测试 7](#_Toc27211)

[6、 小组分工 7](#_Toc14488)

# 实验要求

实现一个提供网上购书功能的网站后端。

网站支持书商在上面开商店，购买者可以通过网站购买。

买家和卖家都可以注册自己的账号。

一个卖家可以开一个或多个网上商店，

买家可以为自已的账户充值，在任意商店购买图书。

支持 下单->付款->发货->收货 流程。

1.实现对应接口的功能，见项目的doc文件夹下面的.md文件描述 （60%）

其中包括：

1)用户权限接口，如注册、登录、登出、注销

2)买家用户接口，如充值、下单、付款

3)卖家用户接口，如创建店铺、填加书籍信息及描述、增加库存

通过对应的功能测试，所有test case都pass

2.为项目添加其它功能 ：（40%）

1)实现后续的流程

发货 -> 收货

2)搜索图书

用户可以通过关键字搜索，参数化的搜索方式；

如搜索范围包括，题目，标签，目录，内容；全站搜索或是当前店铺搜索。

如果显示结果较大，需要分页

(使用全文索引优化查找)

3)订单状态，订单查询和取消定单

用户可以查自已的历史订单，用户也可以取消订单。

取消定单可由买家主动地取消定单，或者买家下单后，经过一段时间超时仍未付款，定单也会自动取消。

# 项目运行

# 数据库设计

# 功能实现

## 4.1 用户功能

注册功能是在原有代码上进行修改的。一共有11个内置函数。

### 4.1.1 token与jwt

可以看到在用户功能板块，利用了token和jwt来确保用户的账户安全。

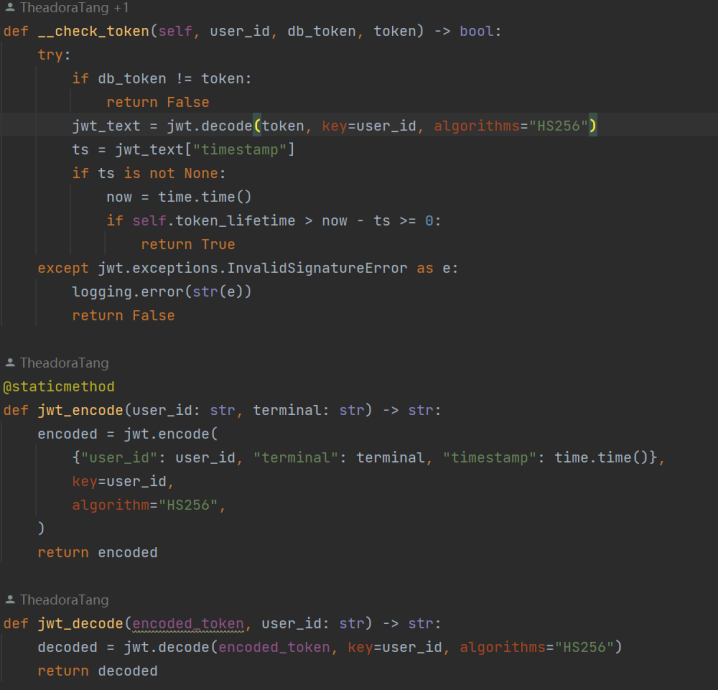
Token，即令牌，是一种用于验证用户身份和授权的技术。 token可以理解为一个字符串，里面包含了足够的信息，以便于服务器识别和验证用户。 token的基本原理是，服务器会为每个已验证的用户生成一个唯一的、有时效性的令牌。 当用户进行一些需要认证的操作时，他们需要携带这个token。而jwt也是经常作为一种安全的token使用。

下面的函数分别用于：

Check\_token:用来检测用户的token，识别和验证用户信息。

Jwt\_encode:用来进行jwt加密。

Jwt\_decode:用来进行jwt解密。

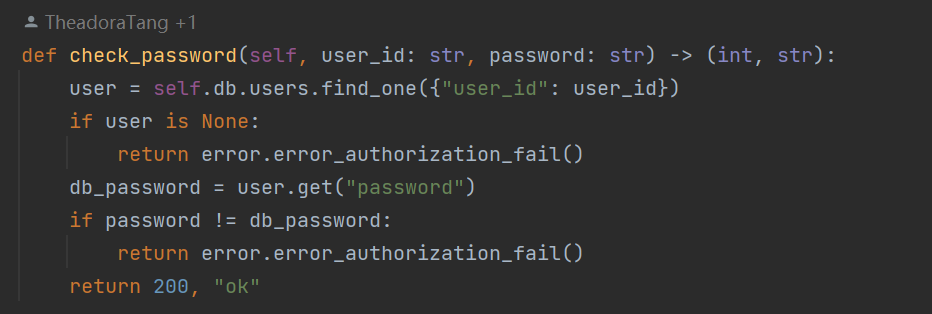


### 4.1.2 register登录



首先检测用户是否存在，如果已经存在，根据error中的定义返回错误的code和message，如果用户不存在，我们将用户的信息存储到bookstore数据库的users collection中，同时返回code和message。

### 4.1.3 check password检查密码



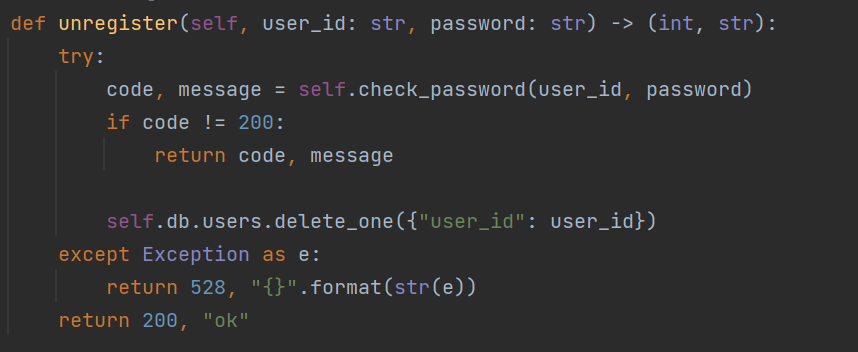
该函数是用来检测用户的密码是否正确，如果不正确返回对应error的code和message，如果正确的话，返回200。

### 4.1.4 logout登出



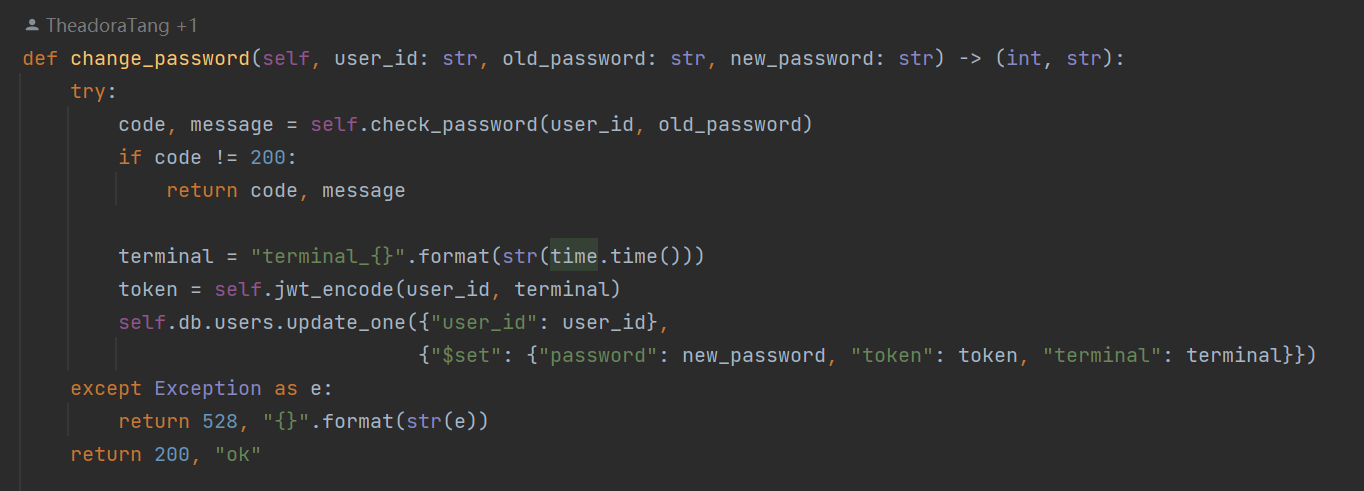
首先利用chech token进行服务器识别和验证用户，然后更新terminal和token，填入到数据库中。

### 4.1.5 unregister注销



首先检测用户输入的密码是否正确，如果验证用户身份成功，则从users中删除有关该用户的所有信息。

### 4.1.6 change password更改密码



首先检测用户输入的密码是否正确，如果验证用户身份成功，则在users的数据原有基础上对password进行更新（terminal和token也同时更新）。

## 4.2 卖家功能

## 4.3 买家功能

## 4.4 发货收货

## 4.5 搜索图书

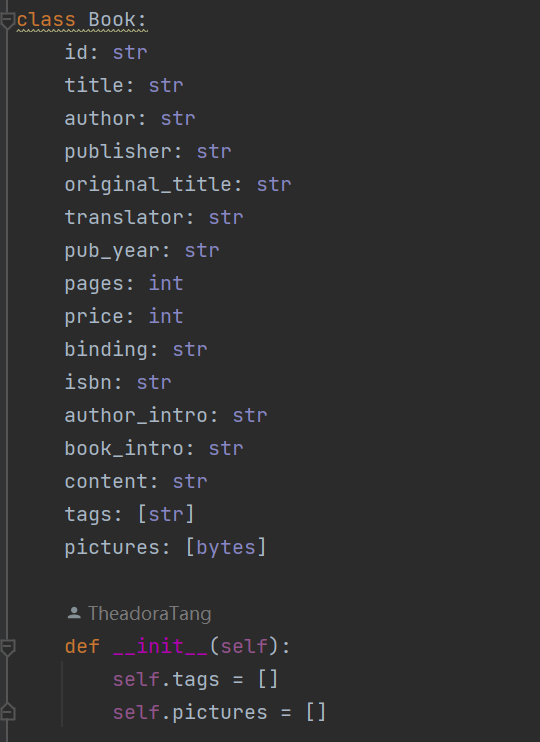
首先，将book\_lx.db中的信息存储进数据库中，我们在init.py中进行存储的操作。

为了将数据从SQLite数据库中提取，并插入到MongoDB数据库中。我们进行了如下的操作：

1. Book类：

用于表示书籍信息。它包含了书籍的各种属性，例如id、title、author、publisher等等。这些属性对应了书籍的各种信息。

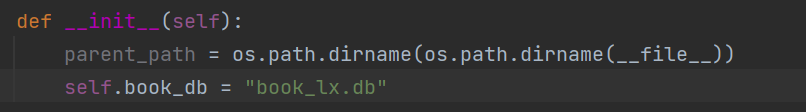
\_\_init\_\_ 方法用于初始化一个Book对象。



2.BookDB类：

用于与SQLite数据库交互以获取书籍信息。

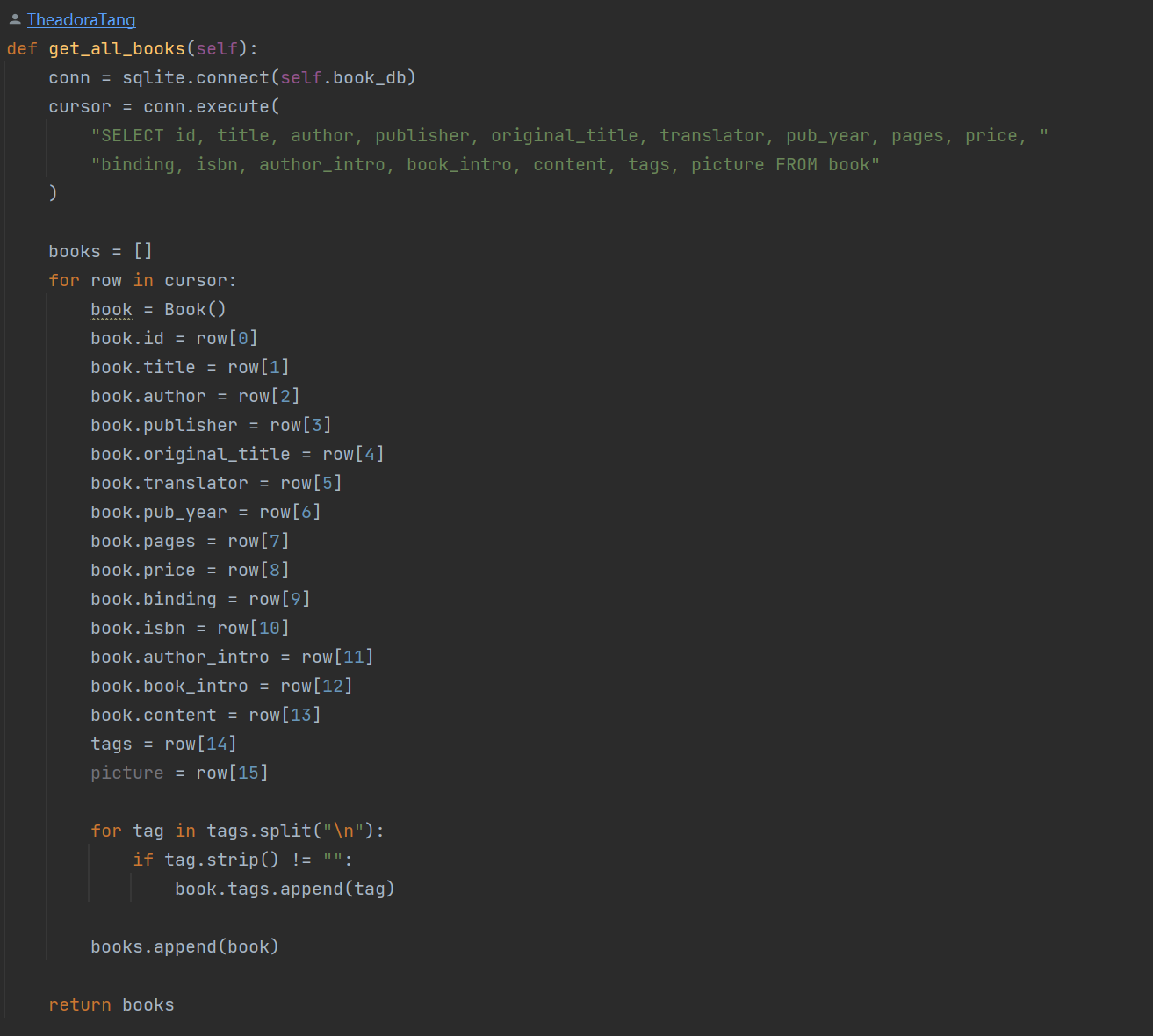
\_\_init\_\_方法初始化了SQLite数据库的路径。



3.\*get\_all\_books方法：

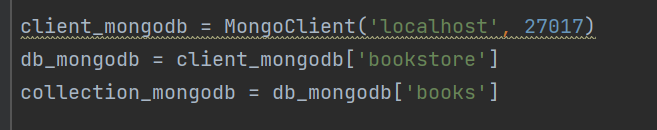
`get\_all\_books` 方法用于从SQLite数据库中提取所有书籍信息。

连接到SQLite数据库，执行一个SQL查询，然后将查询结果转化为Book对象的列表。



1. 连接到MongoDB：

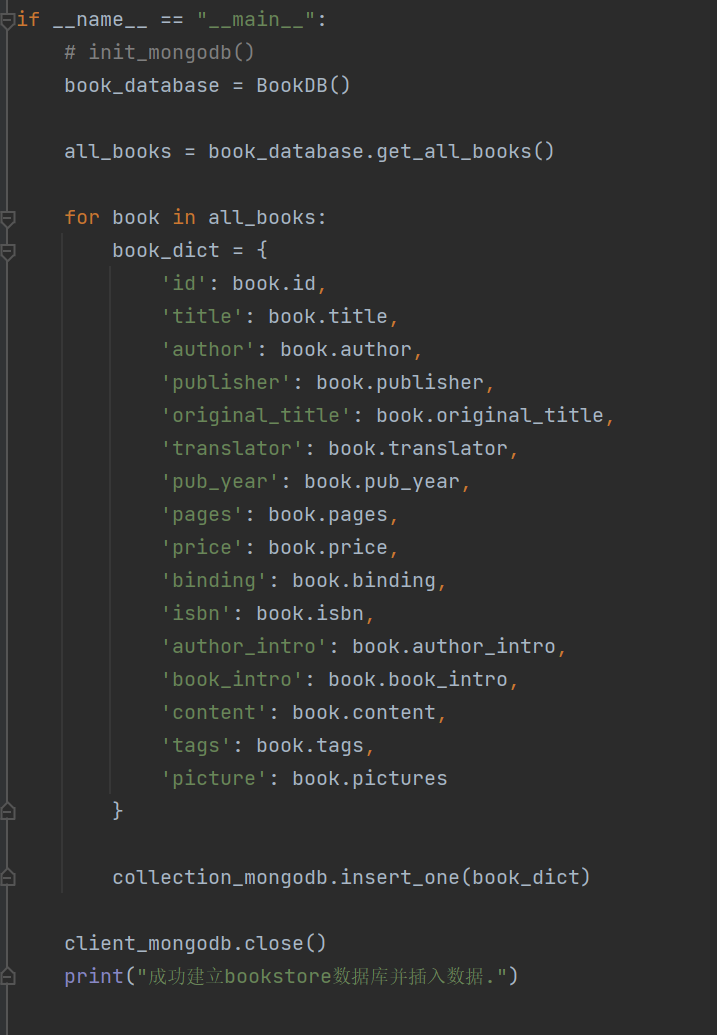
创建了一个MongoDB的客户端，连接到'bookstore'的数据库，并获取名为books的collection。



5. 主程序：

`book\_database = BookDB()` 创建一个BookDB对象，用于与SQLite数据库交互。

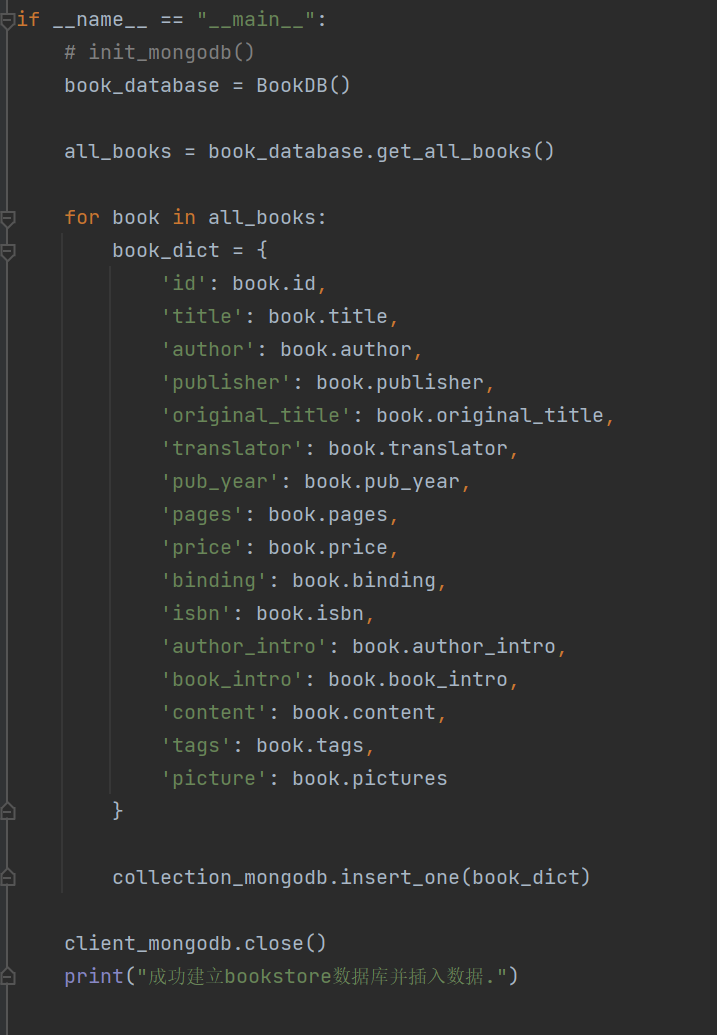
`all\_books = book\_database.get\_all\_books()` 调用`get\_all\_books` 方法获取所有书籍信息，并将其存储在`all\_books`列表中。



6.将数据插入MongoDB：

使用一个循环遍历`all\_books`列表，将每本书的信息存储在一个字典`book\_dict`中。

最后，使用`collection\_mongodb.insert\_one(book\_dict)` 将每本书的字典插入到MongoDB的'books'集合中。



1. 关闭MongoDB连接：使用`client\_mongodb.close()`关闭了MongoDB的连接。

接下来，实现搜索功能

## 4.6 订单状态

# 测试

# 小组分工