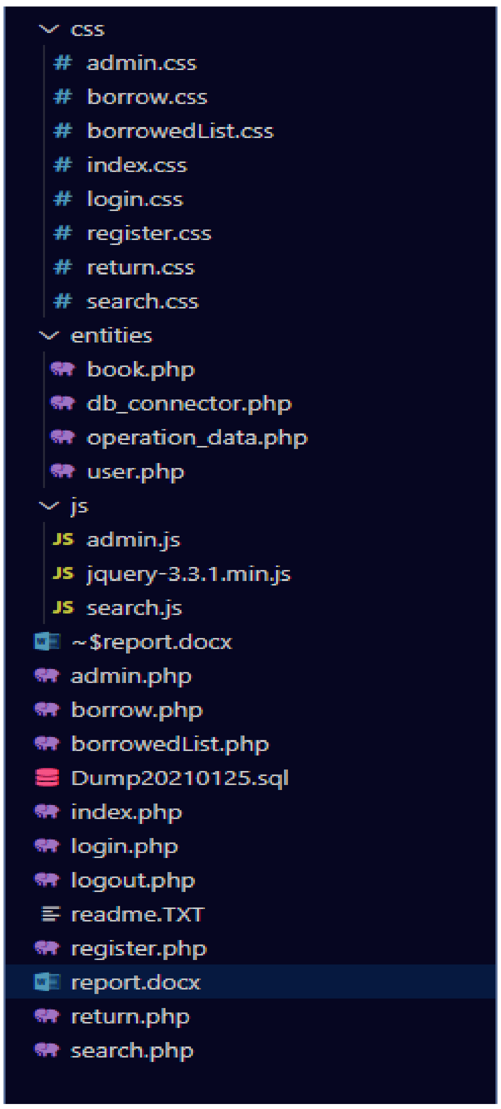
1. Specification

Our app is about managing the information of borrowing books. It is designed for those people who have the needs to managing information of borrowing books. In this app, the users are divided into two categories, normal user and admin user. The normal user can only borrow and return book. The admin user can manage information of books and add books in addition. One of our aim is to let user can know how this app works easily, so we designed very simple pages and function buttons. Once users see the page, they know all things.

2. Structure

The structure of our project is shown in figure 1.



```
css
  # admin.css
  # borrow.css
  # borrowedList.css
  # index.css
  # login.css
  # register.css
  # return.css
  # search.css
entities
  book.php
  db_connector.php
  operation_data.php
  user.php
js
  JS admin.js
  JS jquery-3.3.1.min.js
  JS search.js
~$report.docx
admin.php
borrow.php
borrowedList.php
Dump20210125.sql
index.php
login.php
logout.php
readme.TXT
register.php
report.docx
return.php
search.php
```

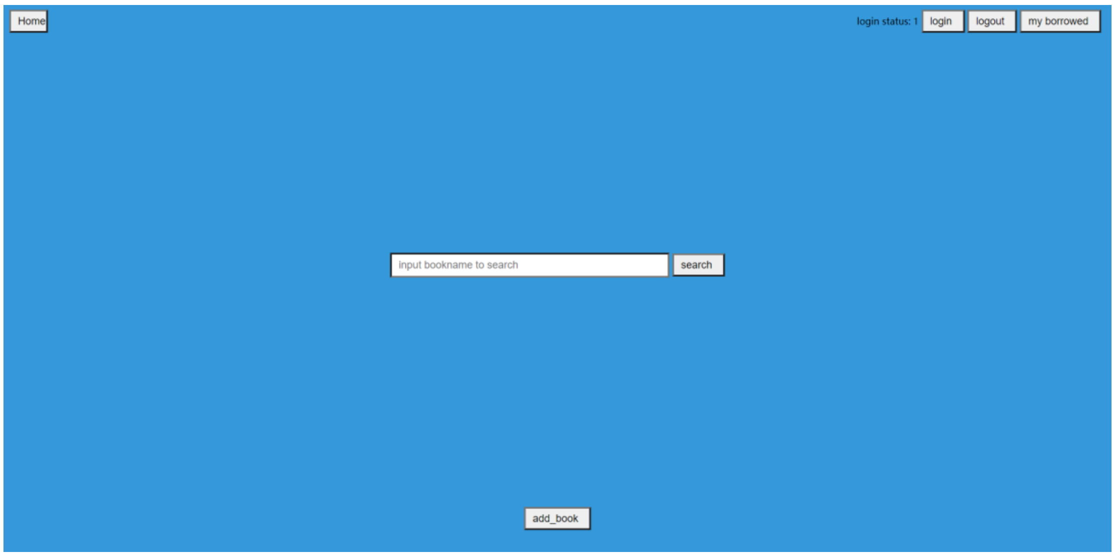
Figure 1

Pages are all shown in php file form. The codes in php file can get data from database and output as html contend form. The css files are used to decorate the contend to html. The js files are used to control the book number information when admin user modifies books. Under the entities category, the four classes represent the data and database operations. The book and user are encapsulated as book class and user class. In 'db_connector.php' file, we can set change parameters that needed to connect MYSQL database. In 'operation_data.php' file, all operations to database are encapsulated as functions of operation class.

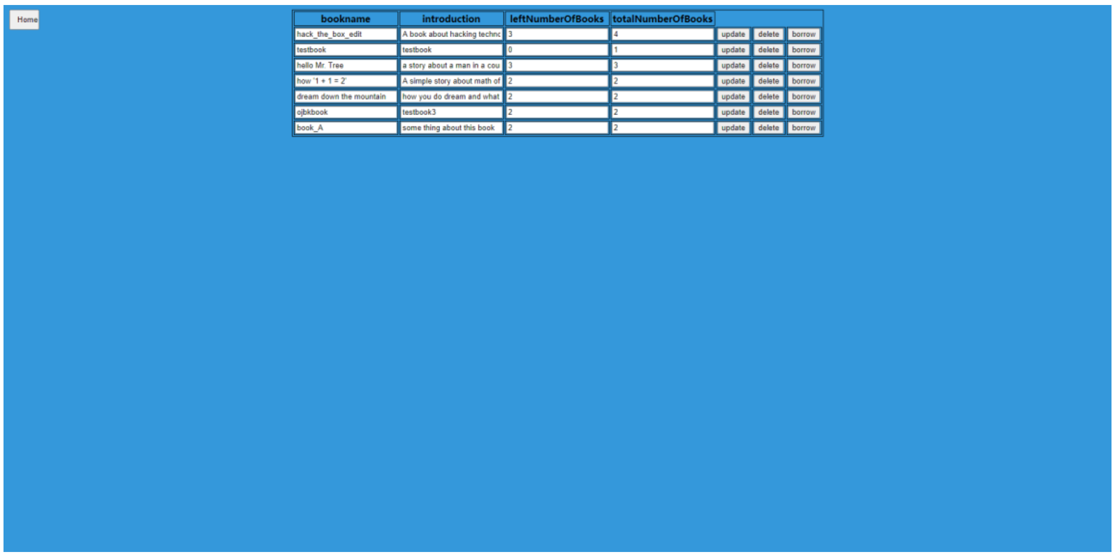
3. Design

If recording information of borrowing book by hands, it is easy to forget something and record wrong data. However, if done by machine, it is much more feasible. In order to realize this purpose, we design an app to manage information of books.

Login as admin user. In contrast with normal, admin can manage books information and add books, which means normal people can't see add book button.

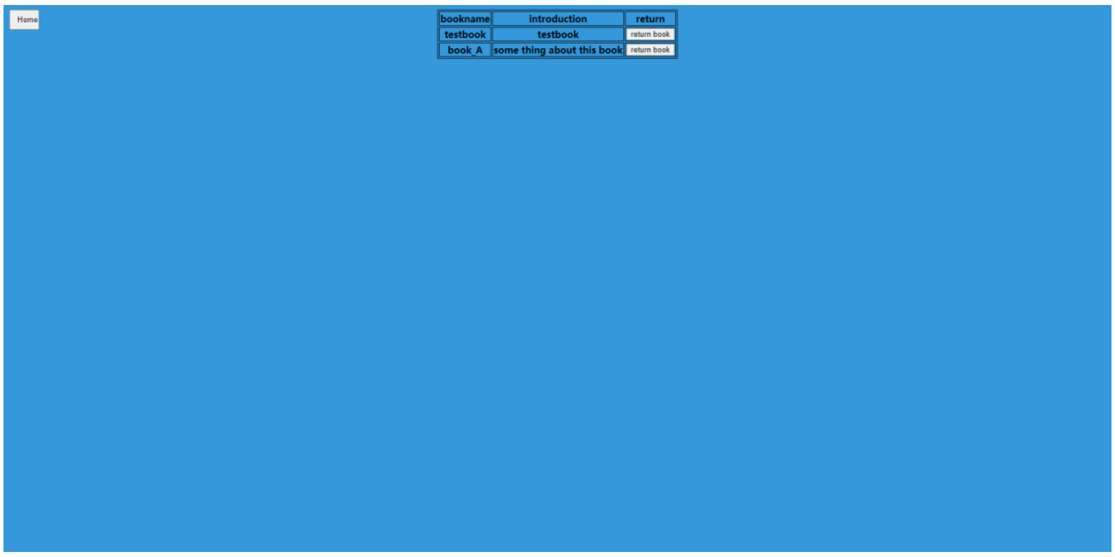


Search result page. Function looks for books according to input string pattern. Users can click borrow button to borrow this book. In also, admin user can modify the book such as updating book and deleting book.



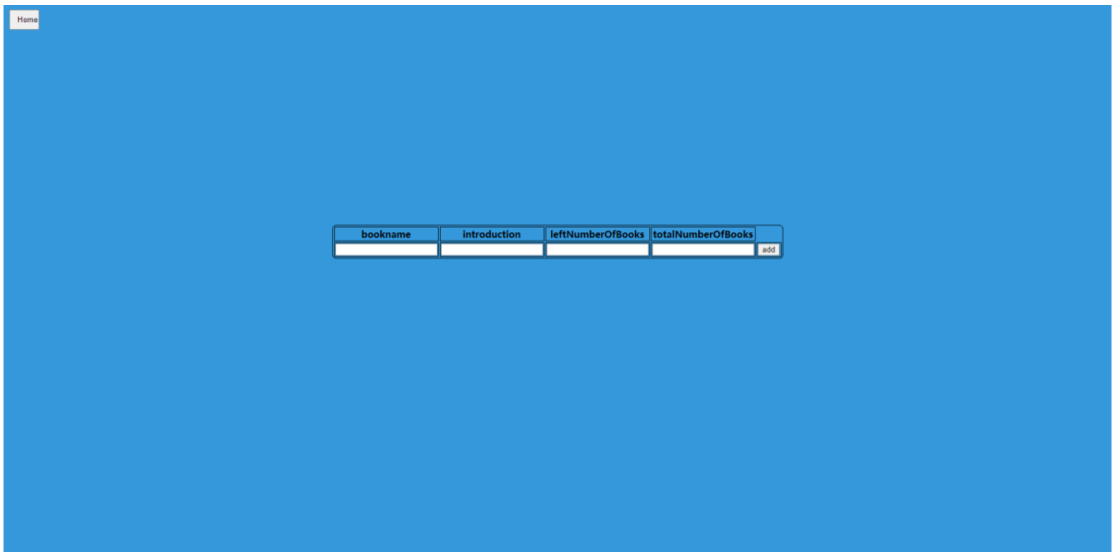
bookname	introduction	leftNumberOfBooks	totalNumberOfBooks			
hack_the_box_edit	A book about hacking technc	3	4	update	delete	borrow
testbook	testbook	0	1	update	delete	borrow
hello Mr. Tree	a story about a man in a cou	3	3	update	delete	borrow
how '1 + 1 = 2'	A simple story about math of	2	2	update	delete	borrow
dream down the mountain	how you do dream and what	2	2	update	delete	borrow
ojbkbook	testbook3	2	2	update	delete	borrow
book_A	some thing about this book	2	2	update	delete	borrow

User can see the books he had borrowed in 'borrowedList' page and can return the book.



bookname	introduction	return
testbook	testbook	return book
book_A	some thing about this book	return book

Add book page. Admin user can add book.



bookname	introduction	leftNumberOfBooks	totalNumberOfBooks	
				add

4. Implementation

In order to realize this app, we code the entities class first. After realizing the connection to database and expressing data as entity object. We start to code pages. Finally, we add CSS to decorate pages and JS to modify admin's action when updating book and adding book. In JS part, we use JQuery because it can reduce the number of codes significantly. In our database, there are three tables, users, books and borrow relations. The borrow relation table uses 'userid' and 'bookid' as key.

5. Test

According to our web page functions, we conducted some tests, the specific details are as follows.

5.1 Register

	Test case	Input	Expected result	Actual result
Invalid	Username: null Password: null	Username: Password:	Register failure	Prompt user to enter username
	Username: null Password: not null	Username: Password: 123456	Register failure	Prompt user to enter username
	Username: not null Password: null	Username: test Password:	Register failure	Prompt user to enter password
	Username: existed Password: Compatible password	Username: test1 Password: 123456	Register failure	Refresh and stay on the registration interface
	Username: existed Password: Incompatible password	Username: test1 Password: 111111	Register failure	Refresh and stay on the registration interface
Valid	Username: not null Password: not null	Username: test1 Password: 123456	Register success	Enter the login and registration interface

5.2 Log in

	Test case	Input	Expected result	Actual result
Invalid	Username: null Password: null	Username: null Password: null	Log in failure	Prompt user to enter username
	Username: null Password: not null	Username: null	Log in failure	Prompt user to enter username

		Password: null		
	Username: not null Password: null	Username: null Password: null	Log in failure	Prompt user to enter password
	Username: nonexistent Password: null	Username: null Password: null	Log in failure	Prompt user to enter password
	Username: nonexistent Password: Compatible password	Username: null Password: null	Log in failure	Prompt that the username or password is invalid
	Username: existing Password: Incompatible password	Username: null Password: null	Log in failure	Prompt that the username or password is invalid
Valid	Username: existing Password: Compatible password	Username: null Password: null	Log in success	Enter the home page

5.3 Main functions

	Test case	**Expected result**	**Actual result**
Admin user and normal user operations	Click Home button in home page	Stay on the home page	Refresh the page
	Click Home button in MyBorrowed page	Return to home page	Return to home page
	Click Home button in BookList page	Return to home page	Return to home page
	Click Home button in	Return to home page	Return to home page

	LogIn/Register page		
	Click LogIn button in home page	Enter logIn page	Enter logIn/Register page
	Click LogOut button in home page	Prompt the user to confirm logout	Prompt the user to confirm logout
	Click MyBorrowed button in home page	Enter MyBorrowed page	Enter MyBorrowed page
	Click Search button in home page without input book name	Enter BookList page	Enter BookList page
	Click Search button in home page with existing book name or keyword	Display book information with corresponding keywords	Display book information with corresponding keywords
	Click Search button in home page with nonexistent book name or keyword	Searching failure	Prompt the user that there is no result, user has to return home page
	Click Borrow button in BookList page(if has entered)	Prompt the user to confirm the borrowing and display whether the borrowing is successful. The remaining amount of the book on the borrowing page is reduced, and the personal page shows that the book is borrowed	Prompt the user to confirm the borrowing and display whether the borrowing is successful. If succeed, the left amount of the book on the BookList page is reduced, and the MyBorrowed page shows that the book is borrowed; if the left amount is 0, it prompts the user the book is not available; if the user has borrowed the same book, it prompts the user borrow failed.

	Click ReturnBook button in MyBorrowed page (if has entered and borrowed book)	Prompt the user to confirm the return and show whether it is successful. The remaining number of the book on the borrowing page increases, and the personal page deletes the book	Prompt the user to confirm the return and show whether it is successful. If succeed, user has to return to home page, the left number of the book on the BookList page increases, and the MyBorrowed page deletes the book.
Admin user operations only	Click AddBook button in home page	Enter AddBook page	Enter AddBook page
	Enter AddBook page and click add button without filling the book name	Adding failure	Prompt the user to enter the book title
	Enter AddBook page and click add button without filling the book introduction	Adding failure	Prompt the user to enter the book introduction
	Enter AddBook page and click add button without filling the book amount	Adding failure	Prompt the user to enter the book amount
	Enter AddBook page and click add button with filling the book amount <= 0	Adding failure	Prompt the user the number is wrong
	Enter AddBook page and click add button with filling all the book details	Prompt the user adding successful	Prompt the user adding successful, the new book is added in to bookList
	Enter BookList page and click the update	Prompt the user update success but data does not change	Prompt the user update success but data does not change

	button without changes		
	Enter BookList page and click the update button and change the name of a book to the name of another existing book	Update failure	Prompt the user update failure
	Enter BookList page and click the update button and change the introduction of a book	Adding success	Prompt the user adding success
	Enter BookList page and click the update button and change the number of a book more the total amount or less than 0	Update failure	Prompt the user update failure and the number is invalid
	Enter BookList page and click the delete button	Prompt user to confirm deletion	Prompt user to confirm deletion and show delete success

We tested the web page and it functions, which is based on the normal operations and accidental operations. The reason for choosing these test cases is to test basic functions and prevent unexpected situations. We simulated and tested the usability of the basic functions we provided to users, and also tested some illegal or possible errors, such as the user did not enter the required value, and the administrator accidentally clicked the delete book button.

6. Evaluation

In our working process, we nearly don't have any big difficulties. One of the reason is our app is much simple – only managing information. And the simple property is exactly one drawback of our app. The data is shown in table form so that user can see clearly. But on the other hand, it lacks of diversity. On the next time, we are going to enrich the book class, so that it can show picture.

Our app is created by all of us. The job division is shown in table 1. It shows which part is mainly done by which person.

	CSS	JS	Entities class	Admin.php	Index.php, Search.php, Return.php	Borrow.php, borrowList.php	Register.php, login.php, logout.php
Lishun cai			✓	✓			
Xiaozhu li						✓	✓
Raoyuan zhao	✓	✓					
Lin han					✓		