

## **OnlineShop Development Report**

### **Description and requirements**

This application is an online shopping website. This is a business to customer(B2C) platform which means it will sell things directly to customers. A B2C website has frontend page and backend page, frontend page is used for displaying products and backend page is used for database management.

Functional requirements:

Frontend:

- a. Home page: Carousel display, Category display, Advertisements display, Special products.
- b. Products information page: Product main picture display, Product description picture display, Product description, Product configuration information, Product comments.
- c. Checkout page: Shipping address, Product information.
- d. Shipping cart page: Single select and select all, Quantity edit, Delete.
- e. Order generation page
- f. Products category page
- g. Search page
- h. Login and registration page: registration (including mobile registration), login (including mobile login).
- I. Recover password page: Register and retrieve password through email.
- i. User information page: My profile, My order, My comment, My address, Edit password.

Backend:

- a. Administrator management
- b. User management
- c. Slider management
- d. Category management
- e. Category advertisement management
- f. Advertisement management
- g. Products management

- h. System management
- i. Comment management
- j. Address management
- k. Order management

Nonfunctional requirements:

Scalability, maintainability and testability.

### **Development methodology**

There are many development methodologies, but the most commonly used are agile development, waterfall development and iterative development. Waterfall development is a traditional development method. It is the most typical and predictive method which strictly follows the step of pre-planned requirements, analysis, design, coding, and testing. Waterfall development requires that each development period be perfect before proceeding to the next step. Iterative development does not require the tasks of each stage to be perfect, but finish developing the whole project first as soon as possible instead of focusing on the shortcomings during development period. After that, the product will be gradually improved according to customers' feedback. Compared with waterfall development, iterative development is suitable for project where the requirements are not clear, so that when the requirements change during the development process, the impact will be smaller than that in waterfall development. Requirements are usually changing in many projects nowadays, so iterative development has more advantages than waterfall development. Agile is a kind of iterative incremental development method. In agile development, the construction of a software project is divided into multiple sub-projects, and each sub-project is tested and have the characteristics of integration and operation. In other words, a large project is divided into multiple small projects that are interconnected, but can also be run independently, and completed separately. Compared with waterfall development and iterative development, agile development may have more flexibility and stability, less unproductive work, faster high-quality delivery, better development team performance, stricter project management control and faster bug detection. It is difficult to analyze and mine all the requirements in any software project's early stage. Some requirements will gradually appear in the process of design and implementation. Agile development is more adaptable to this change than waterfall development.

In this project, I have chosen Agile as development methodology. The reason why Agile was chosen as this project methodology is that it is a mini project developed by few people which requires high standard on the development progress instead of project details. Agile has paved a way to think of the construction of the whole project instead of the special step which is very important for a small project. Therefore, Agile has been chosen in this project.

### **Development and testing environment**

Hardware:

CPU: intel core i7 /RAM: 8G / Hard disk: 1T

Software:

a. Development:

Operating system: Windows 10

Development Language: php

Development framework: Laravel 6.0

IDE: phpStorm

Database: MySQL 5.6

Version Control: Git

b. Testing :

Operating system: Windows 10

Language: python

Environment: python+unittest+selenium+HTMLTestRunner

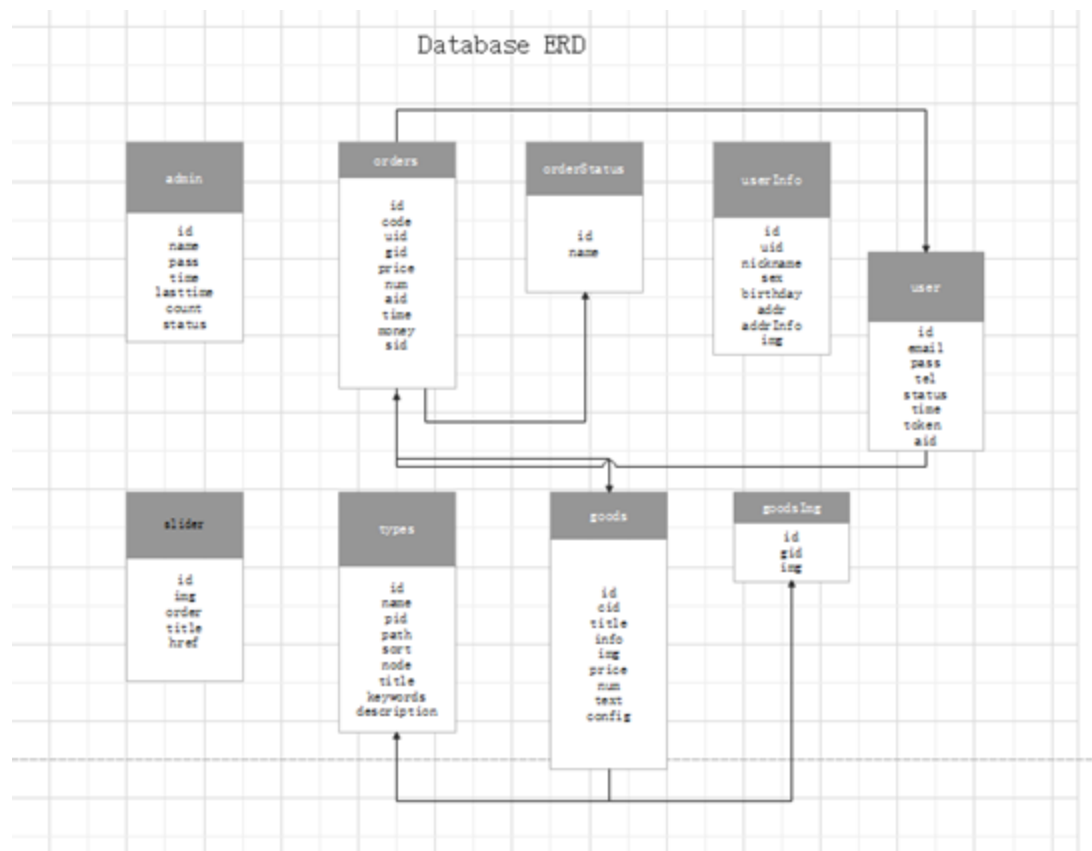
IDE: pycharm

Version Control: Git

### **Development plan and approach**

The development adopts the Laravel framework—an open source web framework based on php language, using the MVC architecture model. The biggest feature and excellence of the laravel framework is the collection of relatively new features of php, as well as various design patterns, Ioc patterns, dependency injection, etc. The features of Laravel are: 1. Powerful rest router: can be called with a simple callback function, quickly bind controller and router. 2. Artisan: command line tool which makes many manual tasks automated. 3. Inheritable templates to simplify the development and management of views. 4. Blade template: faster rendering. 5. ORM operation database. 6. migration: manage database and version control. Therefore, this project chose to develop with laravel.

For a backend management system of online shop, the database design is the first step. According to requirements, there are 13 tables designed, the follows picture shows the ERD of database.



## Testing plan and approach

This project will use black-box testing. The reason why white-box unit test is not used in this project is that although MVC design pattern has been used in development of this project, the model part has been written in controller. It is impossible to test the code coverage of each function except the API. Therefore, in this project, the mainly testing methods will be browser testing.

Black-box testing:

Black-box testing is to test software from outside, it is also called function testing. The following paragraph will demonstrate several black-box testing methods that will be used in this project.

- a. Equivalence class testing. This method does not consider the internal structure of software at all, and only divides the program input field into several parts according to the requirements and descriptions of the software, that is, the requirements specifications. In each part, select a few representative data as test input. To design test cases using the equivalence class division method, equivalence classes must be divided on the basis of analyzing the requirements specification. The so-called equivalence class refers to a certain subset of the input domain, and all the equivalence classes are the entire input domain, which has two important meanings for the test: completeness and no redundancy. Because the equivalence class is determined by the equivalence relationship, the elements in the equivalence class have some characteristics: if an element in the equivalence class is used as test data for testing and the failure in the software cannot be found, then the equivalence class will be used. The other elements of the test are also impossible to find failures. If the test data are all selected from the same equivalence class, except that one of the test data is efficient for finding software faults, using other test data to test is futile, it is meaningless for the progress of testing, it is better to test other equivalent elements. There are usually two steps of equivalence class test, one is to determine the equivalence class, and the other is to design the test cases.
- b. Boundary value analysis. People have found from long-term test work experience that a large number of failures often occur on the boundary of the input definition domain or output value domain, rather than internally. In software design and program writing, the input domain boundary or output domain boundary in the specification are often not paid enough attention, so that errors occur. Designing special test cases to verify the

processing near the boundary can often achieve good test results. To design test cases using boundary value analysis methods, first step is to determine the boundary conditions. The basic idea of the boundary value analysis method is to select the value that is exactly equal to, just above or just below the boundary as the test data, instead of selecting the typical value or any value in the equivalence class as the test data.

- c. Decision table test. The decision table usually consists of four parts: condition pile, condition item, action pile and action item. The condition pile lists all the conditions of the problem. Except for certain problems that have specific requirements on the order of the conditions, the order of the conditions listed here is usually irrelevant. The condition item section lists all possible values for the condition given by the condition pile. Action piles give the possible actions that the problem stipulates, and the order of these actions is generally not constrained. The action item indicates the action that should be taken when each group of condition items takes values. The specific value of any combination of conditions and the corresponding action to be performed are called a rule, and a column that runs through the condition items and action items in the decision table is a rule. Obviously, there are as many rules as there are sets of conditions listed in the decision table. If two or more rules in the table have the same action and there is a very similar relationship between the condition items, it is possible to try to merge them. Combined with the triangle problem, the five steps to construct the decision table are as follows: 1. Determine the number of rules. 2. List all condition piles and action piles. 3. Fill in the condition item. 4. Fill in the action items so that you can get the initial decision table. 5. Simplify the decision table.

Testing cases design analysis:

Backend:

id	requirement	quality	module	page	priority	value
F-01	Add administrator	function	Administrator management	Administrator list	high	1.bullet box 2.username and password tested with boundary value 3.submit and reset button

F-02	Edit	function	Administrator management	Administrator list	high	1.bullet box 2.password 3.status choice 4.submit and reset button
F-03	delete	function	Administrator management	Administrator list	low	1.click and redirect
F-04	Status button	function	Administrator management	Administrator list	high	1.click
F-05	Records display	function	Administrator management	Administrator list	low	1.records display
F-06	search	function	User management	User list	high	1.search user by telephone
F-07	Add category	function	Category management	Category list	high	1.jump to specific page 2.submit and reset button
F-08	delete	function	Category management	Category list	low	1.delete and prompt message before delete
F-09	Add branch	function	Category management	Category list	high	1.jump to correct page 2.when it is not root category then can't be added
F-10	Add product	function	Product management	Product list	high	1.jump to correct page 2.category can be chosen according to category table 3.Images can be correctly added 4.submit and reset button
F-11	Order number	function	Order management	Order list	high	1.order number should be correctly

						linked to order information
F-12	Receiver information	function	Order management	Order list	high	1.receiver information should be correctly linked to receiver information
F-13	operation	function	Order management	Order list	high	1.operation should be correctly linked to status edit page 2.status edit page can be edited 3.submit button
F-14	Order status display	function	Order management	Order status list	low	1.status of each order can be displayed properly 2.status of each order can be edited
F-15	System list	function	System management	System configuration	low	1.title,keywords,description, statistic box can be edited 2.image can be uploaded
F-16	Add slider	function	System management	Slider management	low	1.bullet box 2.boundary value for orders 3.image upload 4.submit and reset button
F-17	Add ads	function	System management	Ads management	low	1.jump to correct page



						2.title,href,sort can be edited correctly 3.picture upload 4.submit and reset button
F-18	Add category ads	function	System management	Category ads management	high	1.jump to correct page 2.choose category 3.picture upload 4.submit and reset button
F-19	Backend management	function	Home page	Home page	low	1.logged administrator name display 2.change password 3.logout
F-20	login	function	Login page	Login page	high	1.display login page when don't log in 2.username and password 3.captcha

## Implementation

### Development

Login page:

On this page, username and password should be passed to database for validation. I have used a plugin for captcha here, so the code of captcha will be passed by session function for validation, once the validation has been done, this page will jump to home page.

There is a middleware class for redirecting to login page if another url has been tried before login.

Route:

//login page

```
Route::get('admin/login','Admin\LoginController@index');
```

//captcha

```
Route::get('admin/captcha','Admin\LoginController@captcha');
```

//login

process

```
Route::post('admin/check','Admin\LoginController@check');  
//logout
```

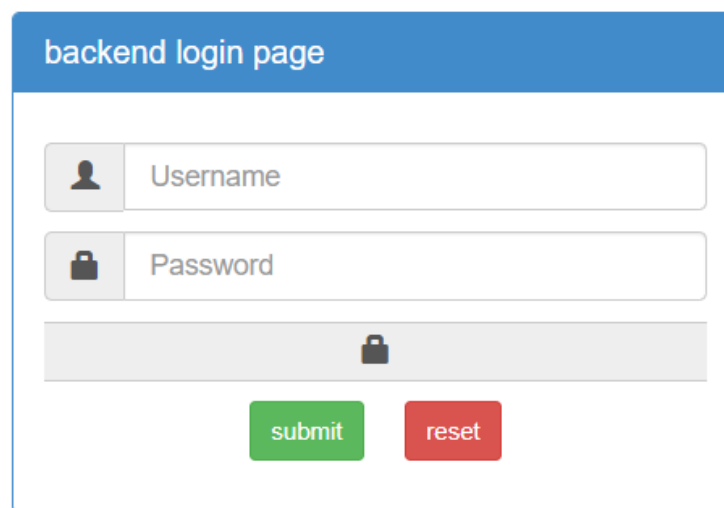
```
Route::get('admin/logout','Admin\LoginController@logout');
```

Controller: LoginController.php

View: login.blade.php

Middleware: adminLogin.php

Plugin: Code.class.php



The image shows a UI mockup for a 'backend login page'. It features a blue header bar with the text 'backend login page'. Below the header, there are two input fields: 'Username' with a user icon and 'Password' with a lock icon. Below these fields is a grey bar containing a lock icon. At the bottom, there are two buttons: a green 'submit' button and a red 'reset' button.

Home page:

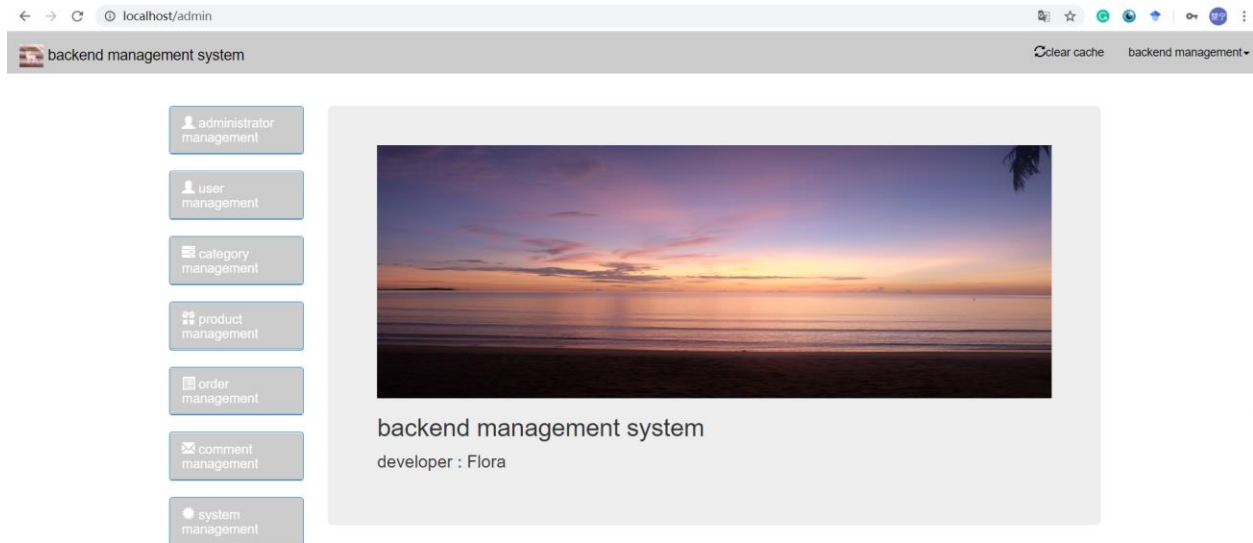
This page has been separated into three parts, the left title bar is the public area so I put this part in a public folder and all the subpages will extend this main page. The right area is only for index url.

Route: Admin route group

Controller: indexController.php

Public view: admin.blade.php

View: index.blade.php



### Administrator management page:

This page provides a view for implementing the table admin in database. There are two bullet boxes, one is for add record and another is for edit record. The password will be encrypted when it is stored to database. There are also limitations for enter name and password, such as no more than 255 characters etc. This page uses 'paginate()' function for page pagination.

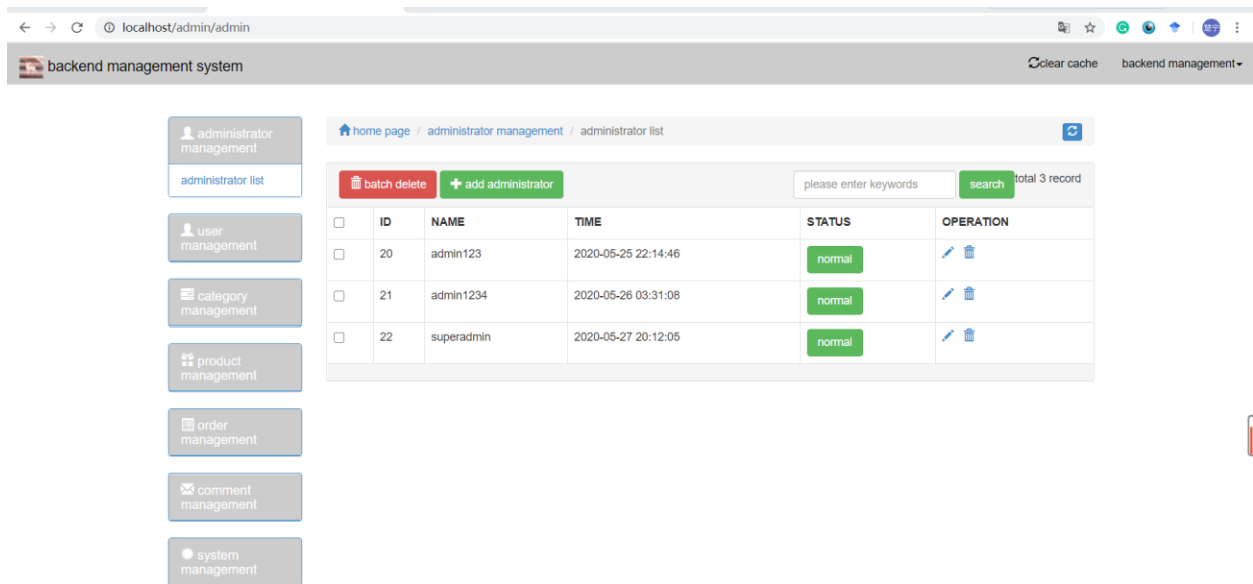
Route:

Route::resource('admin','AdminController');

Controller: AdminController.php

View:resources/views/index.blade.php,

resources/views/add.blade.php, resources/views/edit.blade.php



Add page:

add administrator

user name

please enter user name

password

please enter password

confirm password

please enter password again

status

☒ normal ☐ banned

submit reset

Edit page:

modify information ×

**user name**

admin123

**password**

.....

**confirm password**

.....

**status**

☒ normal ☐ banned

**submit** **reset**

User management page:

This page is for user table. User table stores user account information. The record of this table can be only edited by users.

Route:

Route::get('user','UserController@index');

Controller: UserController.php

View: resources/view/admin/user/index.blade.php

administrator management

user management

user list

category management

product management

order management

comment management

system management

home page / user management / user list

user display

please enter keywords

search

total 6 records

ID	TEL	EMAIL	REGITIME	STATUS
1	213413123	a@qq.com	1970-01-15 00:42:01	not activated
2	2342	211	1980-03-07 23:55:13	activated
3	423423	2133	1970-01-01 00:00:00	black list
42		892865095@qq.com	2017-07-12 12:18:28	activated
44		756388521@qq.com	2020-06-02 02:04:51	activated
56		954407715@qq.com	2020-06-02 08:37:32	activated

### Category management page:

This page is used for implementation of types table. This table has unlimited classification which is a kind of tree data structure. Each row is a type of product, when this type is the parent node of other types, it can be added a child branch. But when one type is the bottom type, it can't be added child branch.

Route:

```
Route::resource('types','TypesController');
```

```
Controller:TypesController.php
```

```
View:resources/views/admin/types/index.blade.php,
```

```
resources/views/admin/types/add.blade.php
```

localhost/admin/types

backend management system

clear cache backend management

administrator management

user management

category management

category list

product management

order management

comment management

system management

home page / category management / category list

batch delete + add category

please enter keywords search total 10 records

ID	NAME	Title	KeyWord	Description	ADD BRANCH	node	OPERATION
1	clothes	clothes	clothesclothes	clothesclothesclothesclothes	add branch	yes	
5	man	man	manman	manmanmanmanman	add branch	yes	
18	accessory	accessoryaccessory	accessoryaccessoryaccessory	121212	add branch	no	
7	shirt	shirtshirt	shirtshirtshirt	shirtshirtshirtshirt	add branch	no	
6	woman	womanwoman	womanwomanwoman	womanwomanwoman	add branch	no	
19	skirt	skirtskirtskirt	skirtskirtskirtskirt	skirtskirtskirtskirtskirt	add branch	yes	
2	electronic	electronic	electronicelectronic	electronicelectronicselectronic	add branch	yes	
9	mobile	mobilemobilemobile	mobilemobilemobile	mobilemobilemobilemobile	add branch	no	
1	cell	cellcellcell	cellcellcellcell	cellcellcellcell	add branch	no	

Add parent type:

localhost/admin/types/create

backend management system

clear cache

administrator management

user management

category management

category list

product management

order management

comment management

system management

home page / category management / add category

category page + add category

category name

please enter category name

TITLE

请输入TITLE

KEYWORD

DESCRIPTION

SORT

is node?

yes no

submit reset

Add child category:

The screenshot shows a web application titled "backend management system". On the left is a sidebar menu with the following items: "administrator management", "user management", "category management" (which is highlighted and contains a sub-link "category list"), "product management", "order management", "comment management", and "system management". The main content area has a breadcrumb trail: "home page / category management / add category". Below the breadcrumb are two buttons: "category page" and "+ add category". The form contains several input fields: "category name" (with placeholder text "please enter category name"), "TITLE" (with placeholder text "请输入TITLE"), "KEYWORD", "DESCRIPTION", and "SORT". At the bottom of the form, there is a section "is node?" with radio buttons for "yes" and "no" (where "no" is selected). Finally, there are "submit" and "reset" buttons.

Product management page:

This page is for implementation of goods and goodsimg table, goods table stores basic information of product and goodsimg table stores image path. I have used a plugin named uploadify for uploading product pictures.

Route:

```
Route::resource('goods','GoodsController');
```

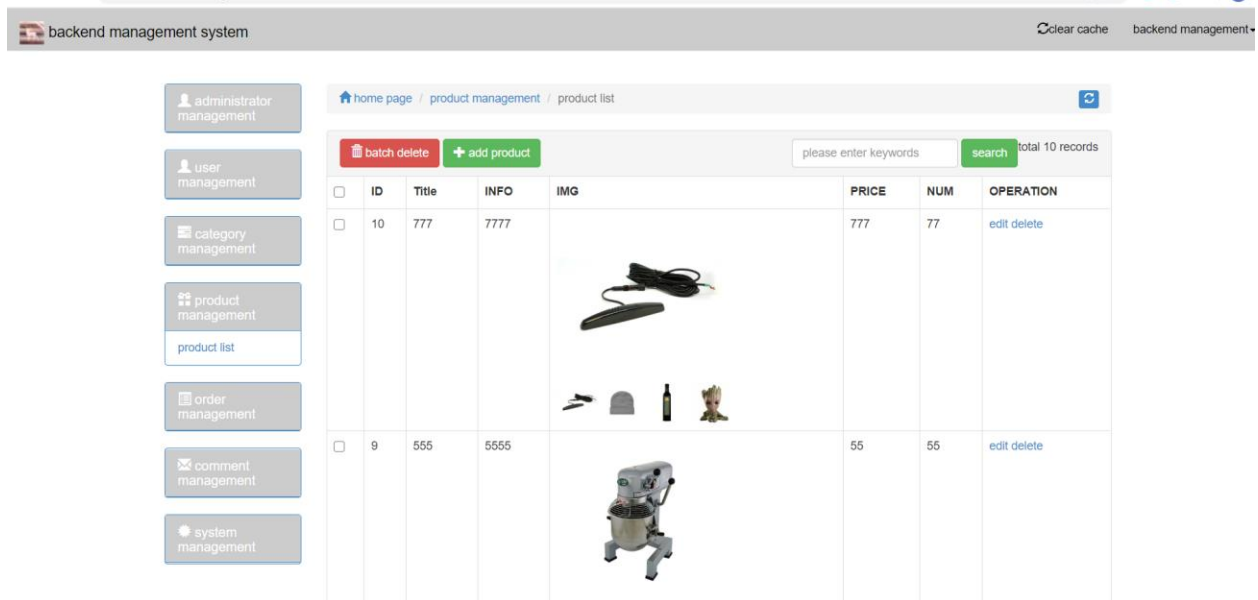
File upload route:

```
Route::any("admin/upload",'Admin\CommonController@upload');
```

Controller: GoodsController.php, CommonController.php

View: resources/views/admin/goods/index.blade.php/add.blade.php





Order management page:

This page is for implementation of orders, orderstatu, goods and goodsimg table. All orders information will be transferred from frontend page.

Route:

// order management

Route::get('orders',"OrdersController@index");

// view order information

Route::get("orders/list","OrdersController@lists");

// view order address

Route::get("orders/addr","OrdersController@addr");

// edit order status

Route::any("orders/edit","OrdersController@edit");

// order status

Route::get("orders/statu","OrdersController@statuList");

Route::post("orders/statu/edit","OrdersController@statuEdit");

View:

backend management system

clear cache backend manager

administrator management

user management

category management

product management

order management

order list

order status list

comment management

system management

home page / order management / order list

order display

please enter keywords

search

total 10 records

order number	user	receiver information	status	order time	operation
0001	user1	receiver information	delivered	1970-01-25 13:13:32	edit status
1100	user3	receiver information	done	1970-01-01 03:05:11	edit status
DZ_149995513511044		receiver information	unpaid	2017-07-13 14:12:15	edit status
DZ_14999553291677		receiver information	unpaid	2017-07-13 14:15:29	edit status
DZ_1499955479928		receiver information	unpaid	2017-07-13 14:17:59	edit status
DZ_14999557428738		receiver information	unpaid	2017-07-13 14:19:34	edit status
DZ_14999557646227		receiver information	unpaid	2017-07-13 14:22:44	edit status

backend management system

clear cache backe

administrator management

user management

category management

product management

order management

order list

order status list

comment management

system management

home page / order management / order status list

order status display

please enter keywords

search

total 6 records

ID	order status
1	unpaid
2	shipped
3	shipping
4	delivering
5	delivered
6	done

## Comment management:

This page is for implementation of comment, goods and goodsim tables. All records will be transferred from frontend page which will show comment under which product with product pictures. The comment cannot be edited.

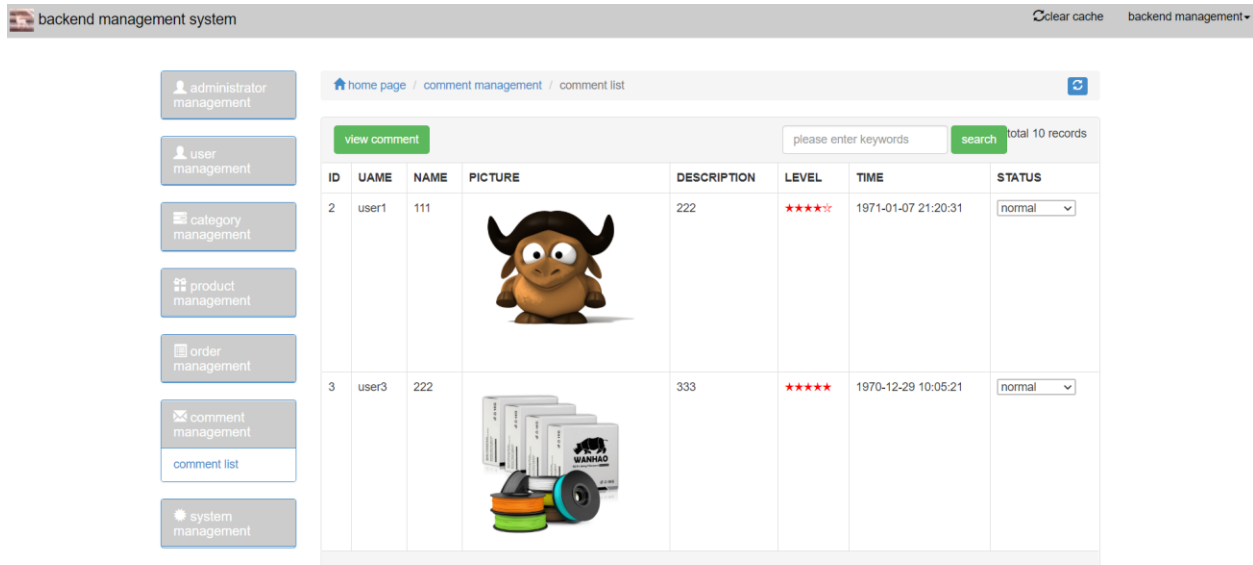
Route:

// comment management

```
Route::get('comment','CommentController@index');
```

```
Route::post('comment/ajaxStatu','CommentController@ajaxStatu');
```

View:



System management:

This page is for implementation of slider, ads and typeads table, all of which will be shown on frontend pages.

Routes:

// system management

// sys management

```
Route::resource("sys/config","ConfigsController");
```

// slider management

```
Route::resource("sys/slider","SliderController");
```

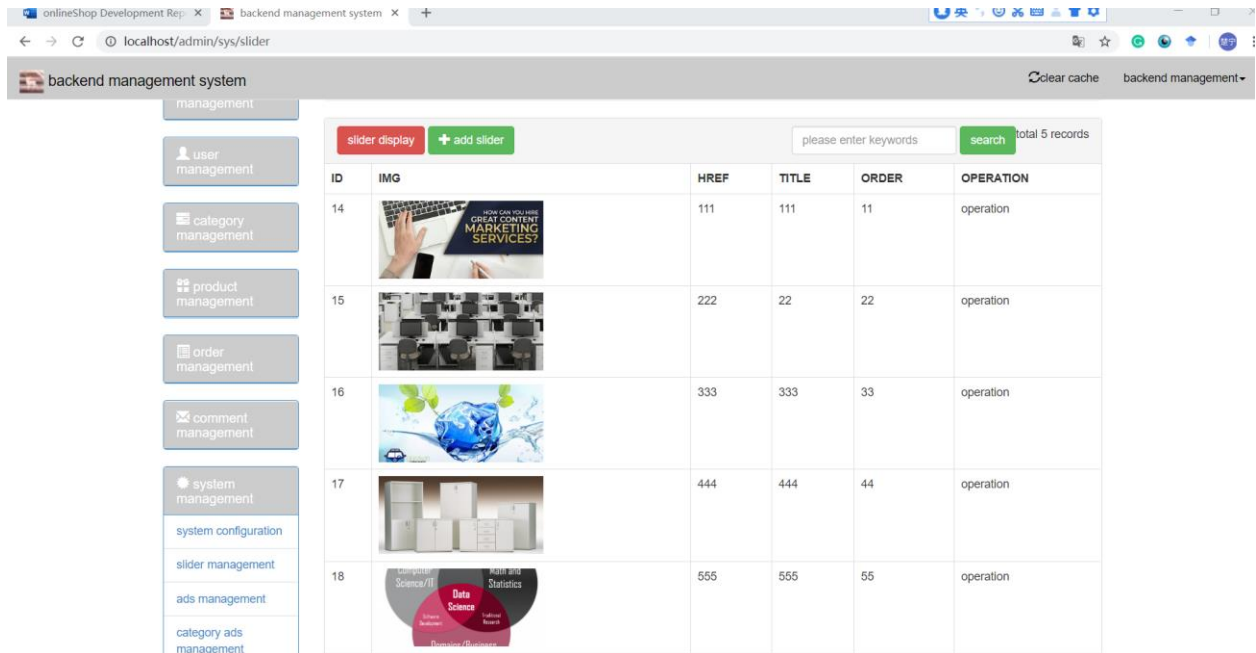
// ads management

```
Route::resource("sys/ads","AdsController");
```

```
// typesAds management
```

```
Route::resource("sys/types","TypesAdsController");
```

View:



## Testing

The testing framework of this project referred to Object Page design model. The directory structure of testing will be shown as follows:

OnlineShopTest/

|----config

|----globalparam.py

|----public

|----common

|----basepage.py

|----log.py

|----mytest.py

- |----publicfunction.py
- |----pyselenium.py
- |----readconfig.py
- |----pages
- |----report
- |----log
- |----testreport
- |----testcase
- |----run.py

Instructions of each file:

globalparam.py: this file encapsulates the method of writing the file to each folder.

Public folder: files in this folder are public modules.

common: files in this folder are some common operations, such as starting the browser.

pages: this folder is used to store page objects of test cases.

basepage.py: this file encapsulated a base page class for page object

log.py: this file is for generating testing logs.

mytest.py: this file encapsulated a base class of start and end for all test cases.

Publicfunction.py: this file encapsulated a function for move screenshots to img folder

Pyselenium.py: this file encapsulates the configuration of the browser driver. The remote browser driver is used here, and the use of the browser can be selected. There are six browsers to choose from. They are Chrome, FireFox, Internet explorer, Edge, opera and phantomjs. This file also encapsulates some basic functions needed for web testing, these functions call the functions that come with python. Therefore, when testing web pages, it is unnecessary to write more operations , just call these functions.

Readconfig.py: read configuration

Testcase: files in this folder are test cases

Run.py: the entrance of the entire program, after testing operation, it will generate html report and log to report and log folder.

Testing running example:

localhost:63342/onlineShopTest/report/testreport/TestResult2020-06-09\_01\_10\_10.html?ijt=c5d59emo777s6sj44m1ve4iaj1

## testing report

tester: Flora  
starttime: 2020-06-09 01:10:10  
totaltime: 0:00:31.528785  
testresult: total 5, passed 3, error 2, passrate= 60.00%

Test the import testcase

summary{ 60.00% } failed{ 0 } passed{ 3 } all{ 5 }

testcases/testcase	total	passed	failed	error	detail
test_admin.TestAdminIndex	3	3	0	0	<a href="#">详细</a>
test_admin2.TestAdminHomePage	1	0	0	1	<a href="#">详细</a>

test\_menu

ft2\_1: 2020-06-09 01:10:32,786 - root - INFO - \*\*\*\*\* START \*\*\*\*\*

2020-06-09 01:10:34,311 - root - INFO - SUCCESS Start a new browser: chrome, Spend 1.5249745845794678 seconds

2020-06-09 01:10:34,495 - root - INFO - SUCCESS Set browser window maximized, Spend 0.1825098991394043 seconds

2020-06-09 01:10:34,826 - root - INFO - SUCCESS Navigated to http://localhost:8080/login, Spend 0.3310508343841553 seconds

2020-06-09 01:10:34,946 - root - INFO - SUCCESS Typed element: <name>username content: superadmin, Spend 0.11768555641174316 seconds

2020-06-09 01:10:35,024 - root - INFO - SUCCESS Typed element: <name>password content: 123456, Spend 0.07770145240783691 seconds

2020-06-09 01:10:35,346 - root - INFO - SUCCESS Clicked element: <name>submit, Spend 0.32114171981811523 seconds

2020-06-09 01:10:35,369 - root - INFO - FAIL Unable to Click by text content: backendmanagement, Spend 0.022938013076782227 seconds

2020-06-09 01:10:37,544 - root - INFO - SUCCESS Closed all window and quit the driver, Spend 2.174039125442585 seconds

2020-06-09 01:10:37,546 - root - INFO - \*\*\*\*\* End \*\*\*\*\*

Traceback (most recent call last):  
File "D:\onlineShopTest\testcase\test\_admin2.py", line 20, in test\_menu  
menu1.click\_dropdown\_link('backendmanagement')  
File "D:\onlineShopTest\public\pages\adminHomePage.py", line 13, in click\_dropdown\_link  
self.dr.click\_text(values)  
File "D:\onlineShopTest\public\common\pyselenium.py", line 304, in click\_text

2020-06-09 01:02:59,719	- root - INFO -	##### START #####	
2020-06-09 01:03:01,585	- root - INFO - SUCCESS	Start a new browser: chrome, Spend 1.864736795425415 seconds	
2020-06-09 01:03:01,849	- root - INFO - SUCCESS	Set browser window maximized, Spend 0.26395368576049805 seconds	
2020-06-09 01:03:02,715	- root - INFO - SUCCESS	Navigated to http://localhost/admin/login, Spend 0.8645925521850586 seconds	
2020-06-09 01:03:02,841	- root - INFO - SUCCESS	Typed element: <name>name content: superadmin, Spend 0.12566375732421875 seconds	
2020-06-09 01:03:02,937	- root - INFO - SUCCESS	Typed element: <name>pass content: 123456, Spend 0.09474682807922363 seconds	
2020-06-09 01:03:03,380	- root - INFO - SUCCESS	Clicked element: <name>submit, Spend 0.44367098808288574 seconds	
2020-06-09 01:03:05,397	- root - INFO - SUCCESS	Get current window title, Spend 0.01455235481262207 seconds	
2020-06-09 01:03:07,631	- root - INFO - SUCCESS	Closed all window and quit the driver, Spend 2.2322113513946533 seconds	
2020-06-09 01:03:07,633	- root - INFO -	##### End #####	
2020-06-09 01:03:07,634	- root - INFO -	##### START #####	
2020-06-09 01:03:09,201	- root - INFO - SUCCESS	Start a new browser: chrome, Spend 1.564051628112793 seconds	
2020-06-09 01:03:09,352	- root - INFO - SUCCESS	Set browser window maximized, Spend 0.14960575103759766 seconds	
2020-06-09 01:03:09,714	- root - INFO - SUCCESS	Navigated to http://localhost/admin/login, Spend 0.3595414161682129 seconds	
2020-06-09 01:03:09,777	- root - INFO - SUCCESS	Typed element: <name>name content: , Spend 0.06183433532714844 seconds	
2020-06-09 01:03:09,838	- root - INFO - SUCCESS	Typed element: <name>pass content: , Spend 0.0608370304107666 seconds	
2020-06-09 01:03:10,114	- root - INFO - SUCCESS	Clicked element: <name>submit, Spend 0.2762629985809326 seconds	
2020-06-09 01:03:12,125	- root - INFO - SUCCESS	Get current window title, Spend 0.00892782211303711 seconds	
2020-06-09 01:03:14,303	- root - INFO - SUCCESS	Closed all window and quit the driver, Spend 2.1773300170898438 seconds	
2020-06-09 01:03:14,304	- root - INFO -	##### End #####	
2020-06-09 01:03:14,305	- root - INFO -	##### START #####	
2020-06-09 01:03:15,914	- root - INFO - SUCCESS	Start a new browser: chrome, Spend 1.608468770980835 seconds	
2020-06-09 01:03:16,064	- root - INFO - SUCCESS	Set browser window maximized, Spend 0.14806389808654785 seconds	
2020-06-09 01:03:16,370	- root - INFO - SUCCESS	Navigated to http://localhost/admin/login, Spend 0.30655479431152344 seconds	
2020-06-09 01:03:16,453	- root - INFO - SUCCESS	Typed element: <name>name content: super, Spend 0.08178496360778809 seconds	
2020-06-09 01:03:16,563	- root - INFO - SUCCESS	Typed element: <name>pass content: 123456, Spend 0.1087045669555664 seconds	
2020-06-09 01:03:16,813	- root - INFO - SUCCESS	Clicked element: <name>submit, Spend 0.2496504783630371 seconds	
2020-06-09 01:03:18,820	- root - INFO - SUCCESS	Get current window title, Spend 0.004988193511962891 seconds	
2020-06-09 01:03:22,207	- root - INFO - SUCCESS	Closed all window and quit the driver, Spend 3.3849270343780518 seconds	

## Limitations

The disadvantage of this mini project is that I used the MVC design pattern in the development process, but I also wrote the code of the model part in the controllers during the actual programming process, which made me unable to do unit testing. In addition, the code of the html page is so simple that there are too few selectors and it is difficult to design test cases.

## References

Waterfall Vs. Agile: Must Know Differences. (2020). Retrieved 9 June 2020, from

[https://www.guru99.com/waterfall-vs-](https://www.guru99.com/waterfall-vs-agile.html#:~:text=Waterfall%20is%20a%20structured%20software,collection%20of%20many%20different%20projects.&text=Agile%20is%20quite%20a%20flexible,initial%20planning%20has%20been%20completed.)

[agile.html#:~:text=Waterfall%20is%20a%20structured%20software,collection%20of%20many%20different%20projects.&text=Agile%20is%20quite%20a%20flexible,initial%20planning%20has%20been%20completed.](https://www.guru99.com/waterfall-vs-agile.html#:~:text=Waterfall%20is%20a%20structured%20software,collection%20of%20many%20different%20projects.&text=Agile%20is%20quite%20a%20flexible,initial%20planning%20has%20been%20completed.)

Differences between Black Box Testing vs White Box Testing - GeeksforGeeks. (2020). Retrieved 9 June

2020, from <https://www.geeksforgeeks.org/differences-between-black-box-testing-vs-white-box-testing/>

Installation - Laravel - The PHP Framework For Web Artisans. (2020). Retrieved 9 June 2020, from

<https://laravel.com/docs/7.x>

chongshi, c. (2016). *selenium2 didonghuaceshishizhan jiyu python yuyan* (1st ed., pp. 289-305). beijing: dianzigongye publisher.