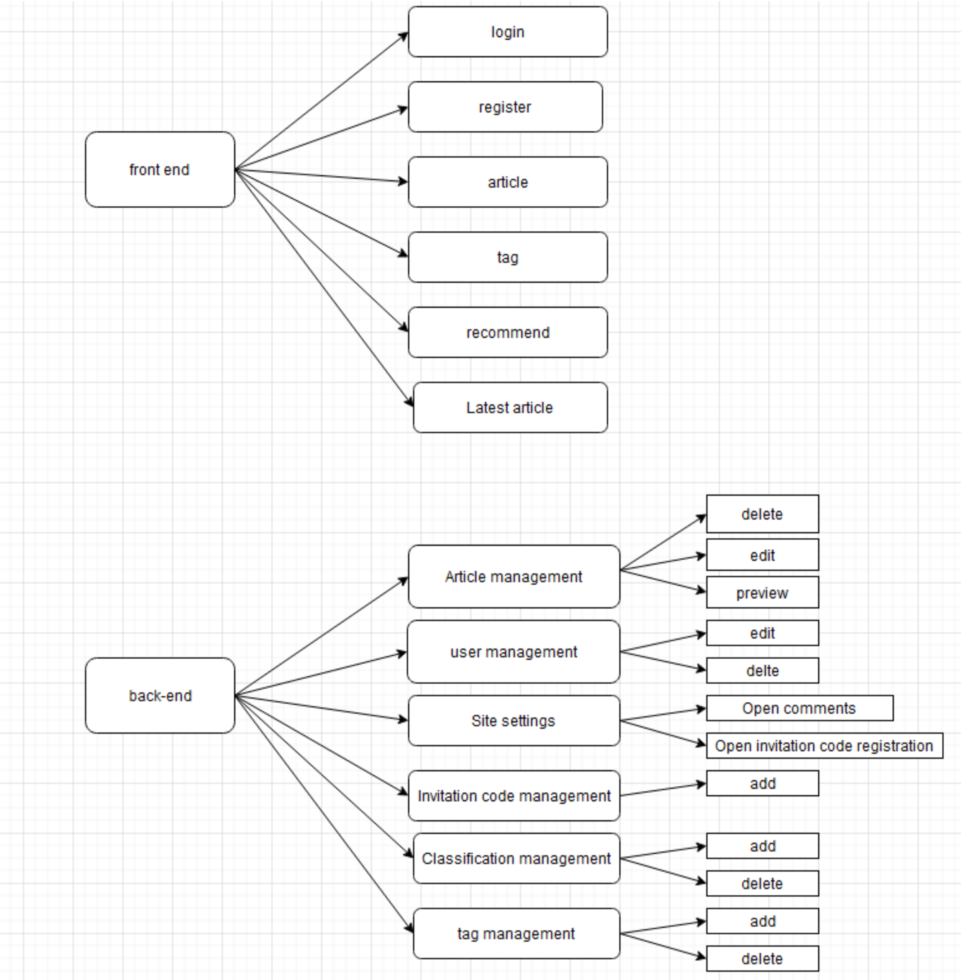
Web Application Development
Coursework 2 -Design Document
2019110011 Fan Dongze

Statement of purpose

The website is a website for sharing novels and comics resources. At the same time, it can also publish some other topics. The website is similar to “masiro” (a website with the same function), but its function is not as comprehensive as ‘’masiro’. In today's subculture circle, it is difficult for people to obtain and publish corresponding resources and discussion space. This website aims to provide entertainment space for small circle culture.



Features implemented

You can only browse website articles when you are not logged in. The login status can publish articles and comments. The administrator has the ability to delete users, give users a new user level, and turn on or off the comment function and invitation code registration function. You can add or modify article classification and article recommendation. On the page, you can view articles by categories and labels, and you can also view the publication order and date of articles by archiving. You can view articles by searching. The article has time ranking and heat ranking. The user can change his password and article content.

Website Deployment

link	Admin username	Admin password
47.99.110.156	fff	fff

Analysis and Evaluation of Architecture

Architecture

This website adopts a three-tier structure

UI (presentation layer): it mainly refers to the interface interacting with users. It is used to receive the data input by the user and display the data required by the user after processing.

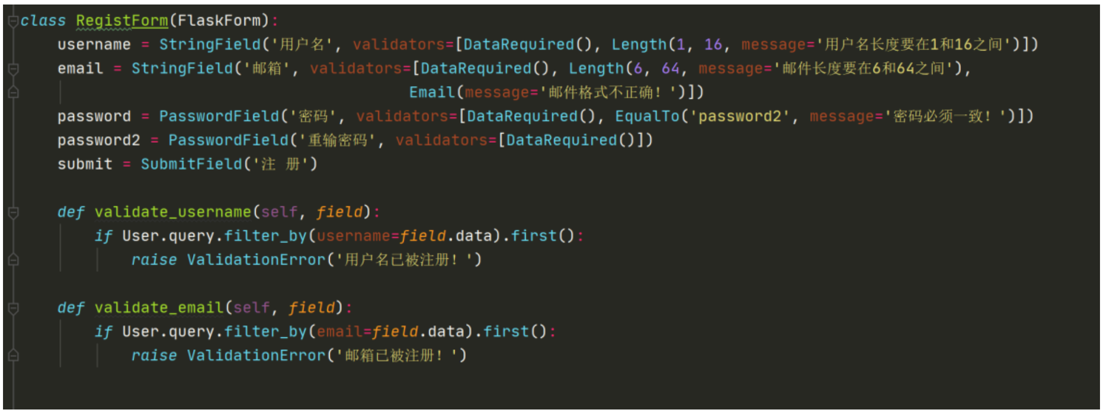
Bll: (business logic layer): the bridge between UI layer and DAL layer. Implement business logic. Business logic includes verification, calculation, business rules, etc.

Dal: (data access layer): dealing with databases. It mainly realizes the addition, deletion, modification and query of data. Submit the data stored in the database to the business layer, and save the data processed by the business layer to the database. These operations are based on the UI layer. The user's needs are reflected in the interface (UI), the UI is reflected in the Bll, and the Bll is reflected in the DAL. The DAL operates the data, and then returns one by one until the data required by the user is fed back to the user

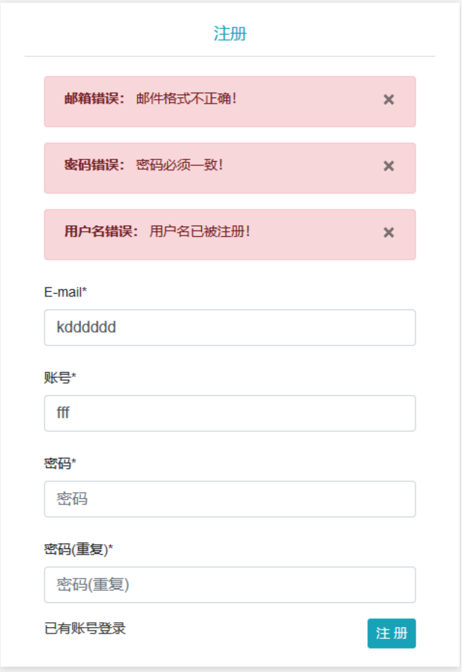
Web forms

The form belongs to the data access layer, which validates the user while entering data.

The website adopts multiple forms, such as user registration, user modification of personal information, article reply, etc. here, take user registration as an example. When wrong information is entered, such as an email address in the wrong format, if the password entered twice does not meet the requirements, it will prompt and require re-entry. In addition to the first time, I also added some function functions to judge whether the user name has been registered and the email address has been registered.



```
class RegistForm(FlaskForm):
    username = StringField('用户名', validators=[DataRequired(), Length(1, 16, message='用户名长度要在1和16之间')])
    email = StringField('邮箱', validators=[DataRequired(), Length(6, 64, message='邮件长度要在6和64之间'),
                                         Email(message='邮件格式不正确！')])
    password = PasswordField('密码', validators=[DataRequired(), EqualTo('password2', message='密码必须一致！')])
    password2 = PasswordField('重输密码', validators=[DataRequired()])
    submit = SubmitField('注 册')

    def validate_username(self, field):
        if User.query.filter_by(username=field.data).first():
            raise ValidationError('用户名已被注册！')

    def validate_email(self, field):
        if User.query.filter_by(email=field.data).first():
            raise ValidationError('邮箱已被注册！')
```



注册

邮箱错误： 邮件格式不正确！

密码错误： 密码必须一致！

用户名错误： 用户名已被注册！

E-mail* kdddddd

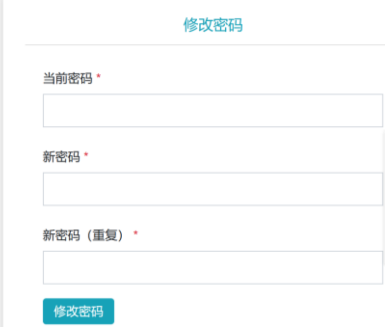
账号* fff

密码* 密码

密码(重复)* 密码(重复)

已有账号登录

In addition, users can also change their passwords



修改密码

当前密码 *

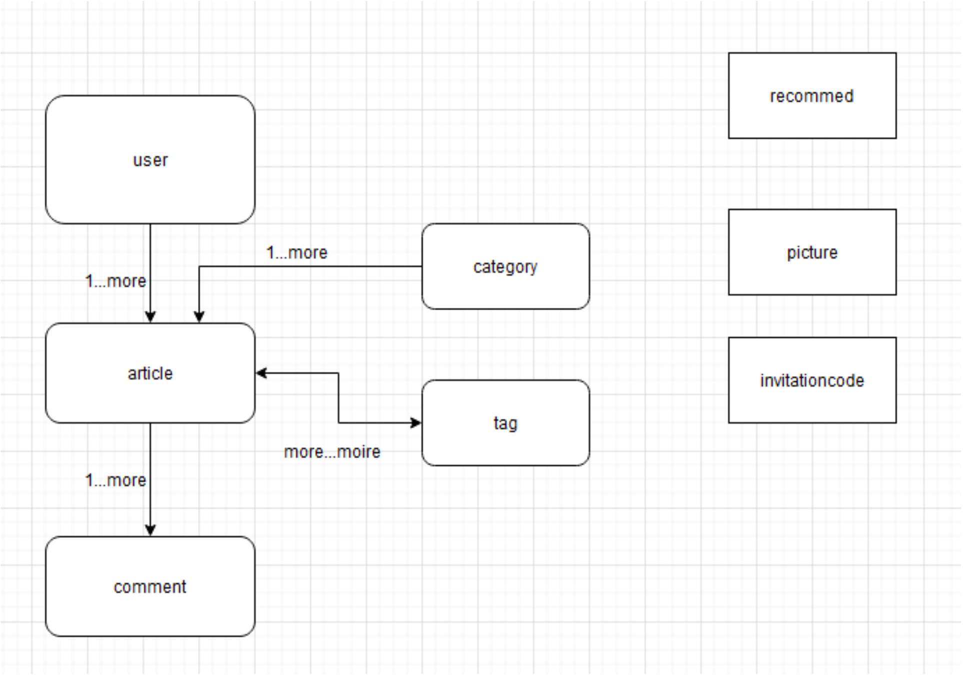
新密码 *

新密码（重复） *

Database

The database belongs to the data access layer and is used to store data.

This website has a total of 8 data models, including users, administrators, categories, labels, articles, comments, recommendations, pictures and invitation codes. A user can create multiple articles. There is a one to many relationship between users and articles, and there is a many to many relationship between articles and labels.

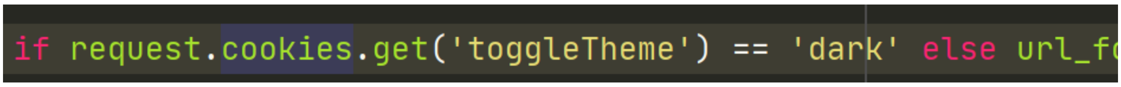


Use of sessions and/or cookies & Authentication

Flask-login is used   to keep the user logged in where session is used.

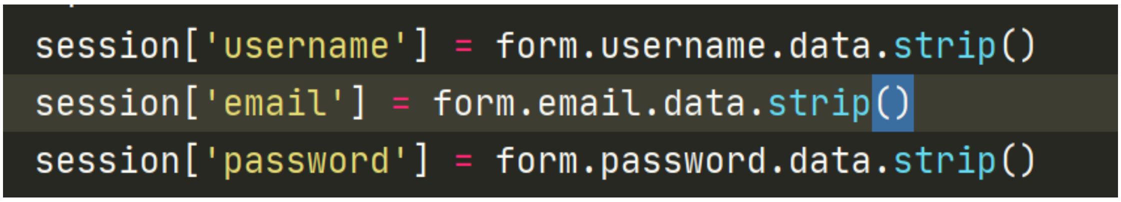
```
from flask_login import login_user, logout_user, login_required, current_user
```

The website creates two website styles according to the user's possible use environment: bright day style and dark night style. The style switching is realized through a button on the index page, and the function uses cookies. The user can also remember the user's previous style selection when closing the browser.



```
if request.cookies.get('toggleTheme') == 'dark' else url_f
```

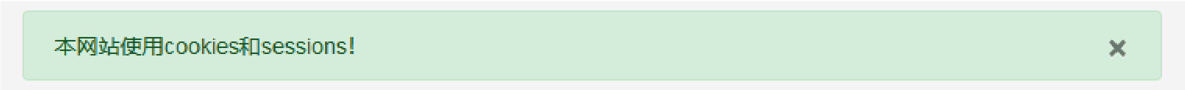
At the same time, this website also uses session. Considering that users may constantly refresh between multiple interfaces, repeatedly connect to the same interface or submit information to one interface, with the record of session, the server can know that this is the same user client completing the operation.



```
session['username'] = form.username.data.strip()
session['email'] = form.email.data.strip()
session['password'] = form.password.data.strip()
```

There is another one named is in flash login_ The authenticated property allows you to see that the user has valid credentials, so you can easily check whether the user has logged in. For example, when browsing a page, you do not need to log in, but when a user prepares to comment or publish an article, you need to judge the user's login status.

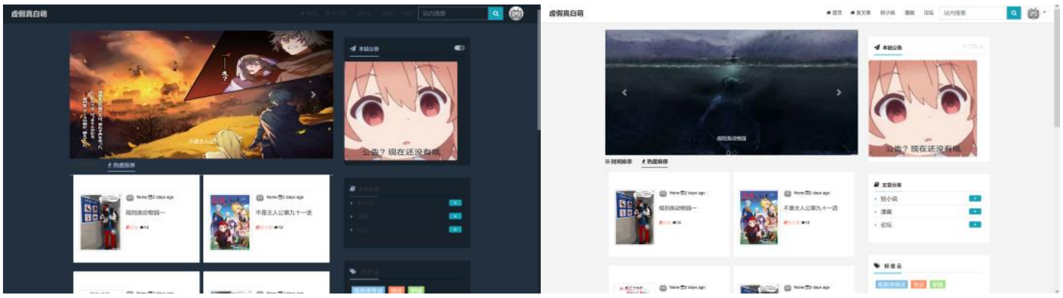
After logging in, this website will pop up a prompt pop-up window for using cookies and session.



本网站使用cookies和sessions!

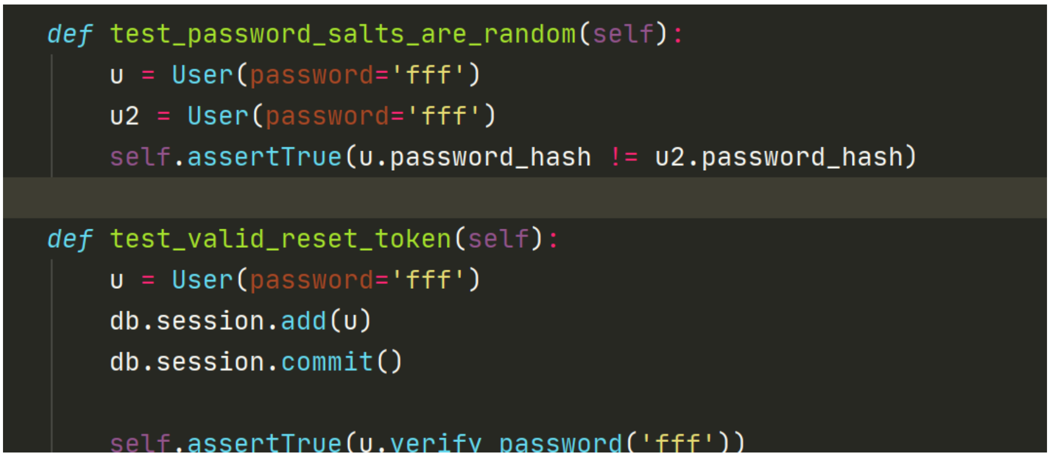
Appropriate styling

This website uses bootstrap framework and template inheritance, so all pages are adaptive. At the same time, for beauty and humanization, the website has two styles of UI



Unit testing

This website has used unit tests to ensure the normal operation of functions, such as password encryption, database submission, the same password is still different after encryption, so as to verify the problems that may not be found in the use of the website. Some codes are as follows:

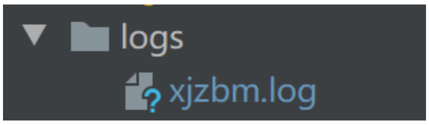


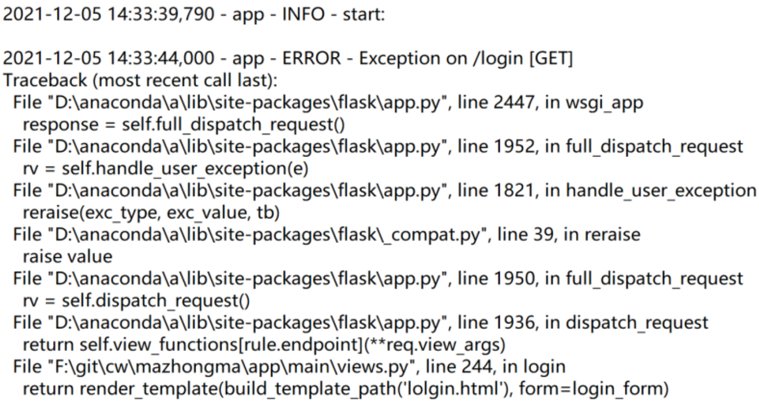
```
def test_password_salts_are_random(self):
    u = User(password='fff')
    u2 = User(password='fff')
    self.assertTrue(u.password_hash != u2.password_hash)

def test_valid_reset_token(self):
    u = User(password='fff')
    db.session.add(u)
    db.session.commit()

    self.assertTrue(u.verify_password('fff'))
```

Logging

```
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')

file_handler = RotatingFileHandler(os.path.join(basedir, 'logs/xjzbm.log'),
                                   maxBytes=10 * 1024 * 1024, backupCount=10)
file_handler.setFormatter(formatter)
file_handler.setLevel(logging.INFO)
console = logging.StreamHandler()
console.setLevel(logging.INFO)
console.setFormatter(formatter)
logging.getLogger("").addHandler(console)
app.logger.addHandler(file_handler)
app.logger.setLevel(logging.INFO)
app.logger.info('start:\n')
```

I use the above code to generate a log, and the generated log will be automatically stored in the file under the log folder.





```
2021-12-05 14:33:39,790 - app - INFO - start:

2021-12-05 14:33:44,000 - app - ERROR - Exception on /login [GET]
Traceback (most recent call last):
  File "D:\anaconda\a\lib\site-packages\flask\app.py", line 2447, in wsgi_app
    response = self.full_dispatch_request()
  File "D:\anaconda\a\lib\site-packages\flask\app.py", line 1952, in full_dispatch_request
    rv = self.handle_user_exception(e)
  File "D:\anaconda\a\lib\site-packages\flask\app.py", line 1821, in handle_user_exception
    reraise(exc_type, exc_value, tb)
  File "D:\anaconda\a\lib\site-packages\flask\_compat.py", line 39, in reraise
    raise value
  File "D:\anaconda\a\lib\site-packages\flask\app.py", line 1950, in full_dispatch_request
    rv = self.dispatch_request()
  File "D:\anaconda\a\lib\site-packages\flask\app.py", line 1936, in dispatch_request
    return self.view_functions[rule.endpoint](**req.view_args)
  File "F:\git\cw\mazhongma\app\main\views.py", line 244, in login
    return render_template(build_template_path('lolgin.html'), form=login_form)
```

Log can record the user's operation errors for better debugging.

Features

I used bootstrap and jQuery framework to help me beautify my page and complete the adaptation of different devices.

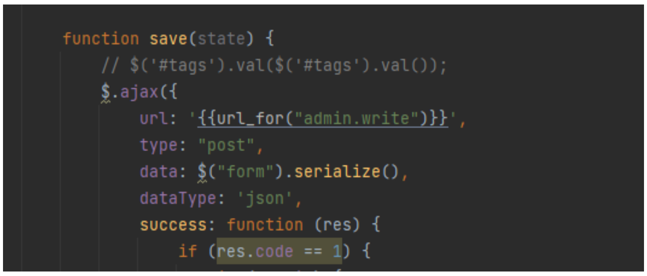
AJAX

This website also uses Ajax features

1) The page does not refresh and update data, which makes the web application respond to user interaction more quickly, avoids sending unchanged information on the network, and reduces user waiting time.

2) Asynchronous communication with the server, without interrupting the user's operation, has more rapid response ability. The communication between browser and server is optimized to reduce unnecessary data transmission, time and data traffic on the network.

3) Front end and back end load balancing: Ajax can transfer the work of some previous servers to the client, use the idle capacity of the client to process, reduce the burden of server and bandwidth, and save space and broadband rental costs. And reduce the burden on the server. The principle of AJAX is "get data on demand", which can minimize the burden on the server caused by redundant requests and responses and improve the site performance.



```
function save(state) {
    // $('#tags').val($('#tags').val());
    $.ajax({
        url: '{{url_for("admin.write")}}',
        type: "post",
        data: $("form").serialize(),
        dataType: 'json',
        success: function (res) {
            if (res.code == 1) {
```

Evaluation

Design decision

Users can publish their own articles on the website, get the resources they want, find the content they need through labels, classification, search and other ways, comment on the content, and delete or modify the content they publish.

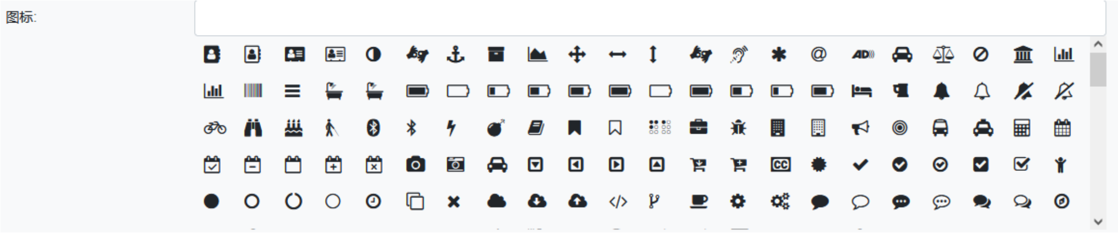
To be improved

Disadvantages of the website: multiple expected functions fail to be realized, such as

Unable to verify the user's mailbox. Send mail to the mailbox address provided by the user.
Unable to grab the words that appear many times in the article, extract keywords and make labels.
Failed to implement the method of becoming VIP user other than using invitation code

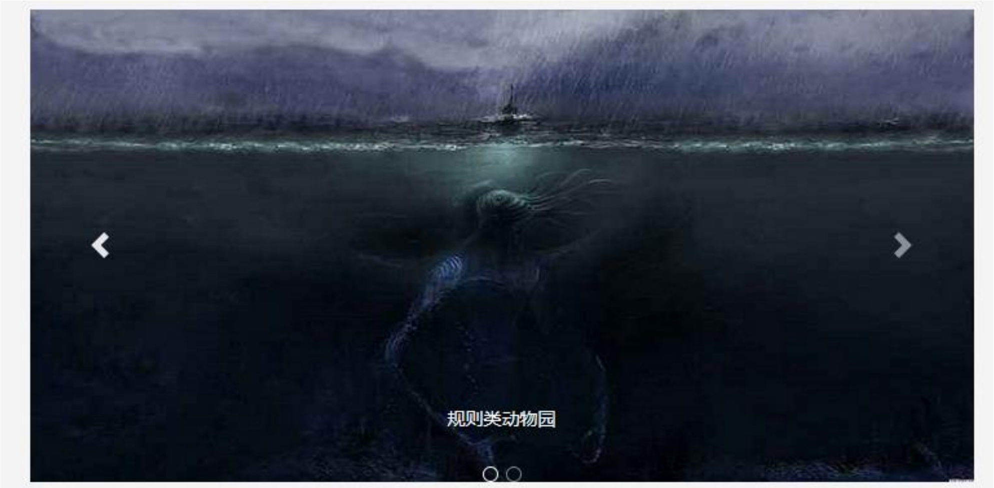
Reached

Website advantages: the use of multiple widgets and the increase of functions, such as
You can add an icon to the category

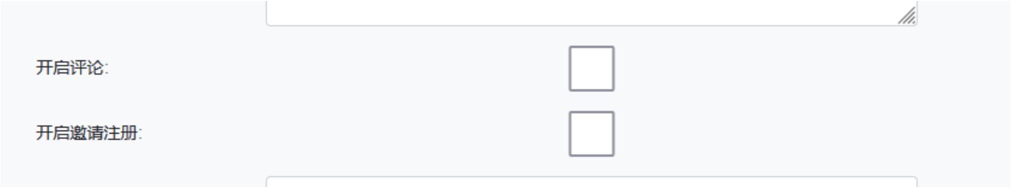


图标:

Invitation code registration can become a VIP member
You can sort the time and popularity, and view the viewing times of the article
You can set specific permissions to access content, such as VIP or administrator
Recommended articles can rotate their own cover (or other customized pictures) on the home page



The function of posting comments and invitation code registration is controlled by the administrator



开启评论: □

开启邀请注册: □

Security issues

CSRF

The full name of CSRF attack is cross site request forgery (Cross Site Request Forgery) is a malicious use of a website. CSRF uses a trusted website by pretending to be a request from a trusted user. What crsf can do includes using your identity to send e-mail, send text messages, conduct transaction transfer, and even steal your account. Of course, there are no transaction transfer operations on this website, but there will still be hidden users exposed Private is possible, so set the secret key to avoid this.

```
SECRET_KEY = os.getenv('SECRET_KEY') or hashlib.new(name='md5', data=b'
python@#').hexdigest()
```

Password security

If the password in the database is stored in clear text, the risk of password theft is increased. Therefore, the password stored in the database should be encrypted. This website only goes to the hash encryption method. To realize this method, we need to call Werkzeug. The password is encrypted through a series of unknown reverse operations. In this way, even if the database password is stolen, the initial password cannot be known, which increases the security of user data.