II .token验证机制

1. jwt说明

参考文章

https://www.ruanyifeng.com/blog/2018/07/json_web_token-tutorial.html

安装authlib模块

pip install authlib –i https://pypi.tuna.tsinghua.edu.cn/simple

```python
from authlib.jose import jwt
#生成token
header = {'alg':'HS256'}
token = jwt.encode(header, payload, key)
#解析token
```

payload中的exp 的失效时间

2. 创建用户表 vul_users

id user_id username password auth create_time status

创建用户模型

通过命令行生成admin账号

```python
@staticmethod
def insert_admin():
    user = VulUser()
    user_id = uuid.uuid4()
    username = 'admin'
    password = 'Com.12Admin'
    auth = 2
    user.user_id = user_id
    user.username = username
    user.password = generate_password_hash(password)
    user.auth = auth
    db.session.add(user)
    db.session.commit()
```

app.py中添加代码

```python
@app.cli.command('create_admin')
def insert_admin():
    VulUser.insert_admin()
```

使用命令flask create_admin 添加管理员账号

2. 编写token.py ,放入lib中

a. 生成token

```python
def generate_auth_token(uid, scope, app, expiration_minutes=60*2):
    header = {'alg':'HS256'}
    expire = datetime.utcnow() + timedelta(minutes=expiration_minutes)
    payload = {
        'uid':uid,
        'exp':expire,
        'scope':scope
    }
    key = app.config.get('SECRET_KEY')
    s = jwt.encode(header, payload,  key)
    return s
```

3. 编写获取 imagecode 代码

a. 安装flask-caching缓存imagecode

pip install flask-caching==2.0.2

使用flask-caching

cache = Cache(config={'CACHE_TYPE': 'simple'})

```
from lib.token import generate_auth_token
from flask_caching import Cache



app = Flask(__name__)
CORS(app)
app.config.from_object(config['development'])
cache = Cache(config={'CACHE_TYPE': 'simple'})
db.init_app(app)
cache.init_app(app)

@app.cli.command('create_admin')
```

b. 编写获取imagecode代码

编写随机生成验证码函数，放入tools.py中

```
def generate_verify_code():
    ls = list(string.ascii_letters + string.digits)
    s = ''
    for i in range(4):
        s += random.choice(ls)
    return s
```

编写获取图形验证码视图函数

```
@app.route('/imagecode', methods=['GET'])
# @cache.cached(timeout=60)
def get_image_code():
    code= generate_verify_code()
    uid = str(uuid.uuid1())
    cache.set(uid, code, timeout=120)
    return {
                'uid':uid,
                'image_code':code
            }
```

c. 编写获取token的视图函数

获取token，需要同时验证图形验证码，用户名，密码。

编写用户名和密码验证代码，以及认证失败代码。

```
class AuthFailed(APIException):
    code = 401
    error_code = 1002
    status = 0
    msg = 'auth failed'
```

用户名和密码认证

```
@staticmethod
def verify(username, password):
    user = VulUser.query.filter_by(username=username).first()
    if user is None:
        return False
    if not user.check_password(password):
        return False
    if user.auth == 1:
        user_scope = 'UserScope'
    if user.auth == 2:
        user_scope = 'AdminScope'
    return {'uid': user.user_id, 'scope': user_scope}
```

编写UserForm

```
class UserForm(BaseForm):
    username = StringField('username', validators=[DataRequired('不允许为空')])
    password = StringField('password', validators=[DataRequired('不允许为空')])
    image_code = StringField('image_code', validators=[DataRequired('不允许为空')])
    uid = StringField('uid', validators=[DataRequired('不允许为空')])
```

编写获取token的视图函数

```python
@app.route('/token')
def get_token():
    form = UserForm().validate_for_api()
    username = form.username.data
    password = form.password.data
    image_code = form.image_code.data
    uid = form.uid.data
    value = cache.get(uid)

    if value is None:
        raise AuthFailed('图形验证码失效')
    if value != image_code:
        raise AuthFailed('图形验证码失效')

    vulUser = VulUser.verify(username, password)
    if not vulUser:
        raise AuthFailed()

    token = generate_auth_token(vulUser['uid'], vulUser['scope'], app)
    ...
    t = {
        'status': 1,
        'code': 200,
        'token': token.decode('ascii'),
    }
    return t
```