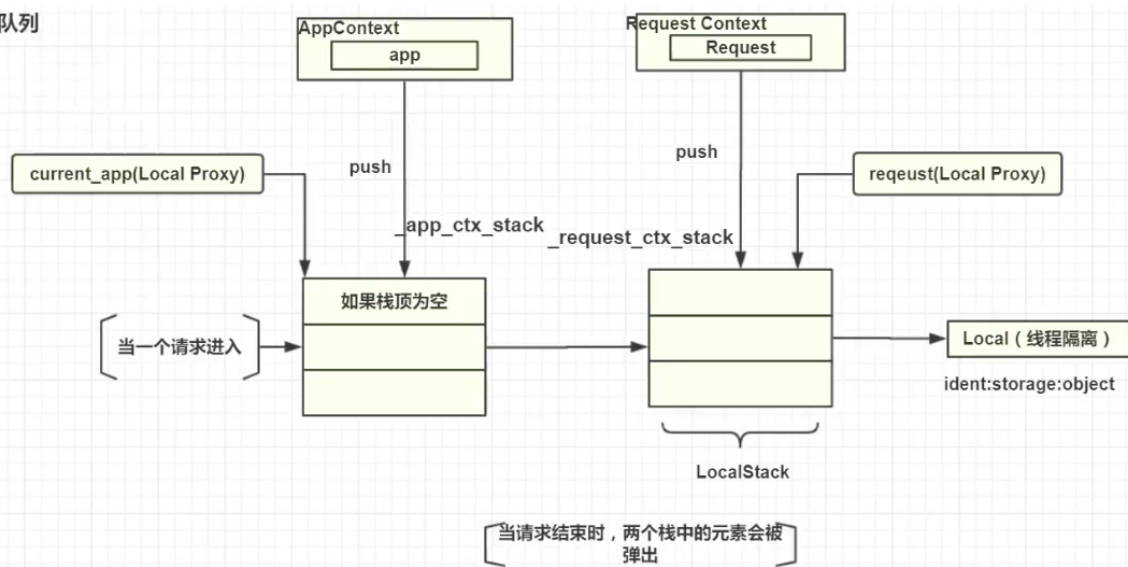生成token：

```
1  from itsdangerous import TimedJSONWebSignatureSerializer as Serializer
2  def generate_token(self, expiration=600):
3    s = Serializer(current_app.config['SECRET_KEY'], expiration)
4    return s.dumps({'id': self.id}).decode('utf-8')
5
6  @staticmethod
7  def reset_password(token, new_password):
8    s = Serializer(current_app.config['SECRET_KEY'])
9    try:
10     data = s.loads(token.encode('utf-8'))
11   except:
12     return False
13   uid = data.get('id')
14   with db.auto_commit():
15     user = User.query.get(uid)
16     user.password = new_password
17   return True
```

```
1  def set_attrs(self, attrs_dict):
2    for key, value in attrs_dict.items():
3      if hasattr(self, key) and key != 'id':
4        setattr(self, key, value)
```

自定义验证器：

validate_字段名

```
1
2  @web.app_errorhandler(404)
3  def not_found(e):
4    # AOP 思想
5    return render_template('404.html'), 404
6
```

栈 队列

AppContext

app

Request Context

Request

current_app(Local Proxy)

push

reqeust(Local Proxy)

push

_app_ctx_stack

_request_ctx_stack

当一个请求进入

如果栈顶为空

Local（线程隔离）

ident:storage:object

LocalStack

当请求结束时，两个栈中的元素会被弹出