7/7/2021, 1/31/2022

4/22/22,/4/24

To be Done

1. Postman
2. Curl
3. **Python 断言和异常处理**
4. Setup and cleanup
5. Json, pickle
6. Generate json test case; analyze json file
7. Csv

<https://blog.csdn.net/wjplearning/article/details/81711032?spm=1001.2101.3001.6650.3&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromBaidu%7Edefault-3.pc_relevant_default&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromBaidu%7Edefault-3.pc_relevant_default&utm_relevant_index=6>

1. API test case
2. Why API test
3. API test workflow

# python使用pytest接口自动化测试的使用

# <https://blog.csdn.net/stetstet/article/details/119242408?utm_medium=distribute.pc_relevant.none-task-blog-2~default~baidujs_baidulandingword~default-1.pc_relevant_default&spm=1001.2101.3001.4242.2&utm_relevant_index=4>

1. API doc
2. What to test
3. Json

# Python操作数据库

IQramp 🡪 csv file for test cases for selenium UI and json for restAPI

1. API test case

1、功能：检查接口基础功能，是否完成了业务逻辑要求。

此处的用例设计方法，和普通的测试用例设计方法一样。可以把接口当作一个待测模块，分析接口功能需求，利用常规用例方法设计测试用例。可供参考的用例设计方法如下：

2、数据：分析接口的输入参数，覆盖各种可能的场景。

（1）检查接口的输入，数据格式、数据类型、数据范围等

（2）检查接口的参数边界（传递的参数足够大或者为负、空值时）

（3）检查接口的参数的组合，可选、必选等

（4）检查接口的约束条件，不符合约束条件的，不需要设计用例

**用例筛选**

此接口，接近30个用例，已经比较全面了，但实际上有点冗余。比如，如果调用接口的界面是个用户选择界面，那么参数上，会受到约束，uid/orderNum都从前端传入，是一些固定的值，那么就不会产生一些特殊情况。类似以下情况，就不会在实际中遇到，因此这些用例的价值趋于0，完全可以在用例文档中去除

设计的用例在考虑全面之后，需要再做一次减法，保证用例的实效性。对无法出现的场景设计出来的用例，毫无价值，只会增加我们的工作量，对产品质量提升没有帮助。因此，测试用例并不是越多越好，在于该用例是否能真正预防bug。（3）中灰底用例去除



1. Why API test

* Reason-1
* API service, public API end points used by customers.
* Private API, efficient way to verified certain features without going through UI or **UI not able** to verify.
* Find bugs that UI can’t
* Check the handling of exception from backend
* Check system’s security and safety



1. API test workflow



* General flowchart-1
* Review API doc

Design input parameters

Get response

Parse response and assert

Determine pass/fail

* Flowchart-2
* Prepare (80%)

Read API doc, understand input parameters and business logic/flow

Create test cases and test data

* Execute test cases in test environment; collect test results (10%)
* Analysis(10%): data/log analysis
* Test report
* Flowchart-3

接口测试需要围绕测试计划实施

接口测试重点是：**api文档分析、用例设计**

接口api文档分析要重点查看下面两项：

1、请求参数说明 如：是否必填、类型、取值范围

2、接口之间业务以来关系

具体接口应用工具主要根据后台协议和测试计划测试工具来定，可使用工具如：postman、jmeter、requests、抓包（fiddler、charles）

**api文档分析**【重点】

该环节主要分析：请求和响应。

**请求**：

1、请求url是否合理

2、请求信息头是否明确

3、请求方法是否合理

4、请求参数是否必填、含义、参数方式（url\form\json）

**响应**：

1、响应状态码是否合理

2、响应数据格式是否明确

3、响应信息头内容是否明确

异常：异常返回信息和错误代码

业务：**接口彼此之间依赖关系**

该流程重点是：

1. 请求**参数方式和类型**  
   2、**接口之间依赖关系**  
   3、错误异常

**流程3、测试计划**

由测试Leader编写，目的：**分配测试任务进度安排和指导测试方案**执行。

核心：测什么、谁来测、怎么测、风险预测及控制; jira**流程**

**用例设计**

提示：该环节主要针对参数、测试数据、安全、依赖、异常进行覆盖。

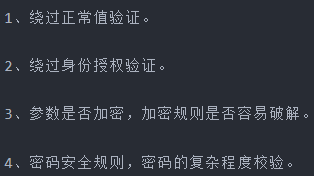
* **通过性验证**：首先保证接口好用，按文档正常传入，查看是否可以返回正确的结果。
* **参数组合**： 按接口文档中对参数的要求进行有目的的组合，比如必填未填是否通过，标志类参数值的切换是否能对应正确的功能等。（这部分很关键

接口调用有两种传参方式：**URL-key-value**形式，**Json串传参形式**。

key-value形式可以把参数拼接在url的后面由?相连，多个参数之间用&相连，如url?parameter1=key1&meter2=key2…

Json串传参不能把参数直接连在url中，需要写在请求的body里面，

* **异常验证**：不按照接口文档上的要求输入参数，来验证接口对异常情况的反应
* **接口安全**



1）、**通用接口用例**设计

①、通过性验证：首先肯定要保证这个接口功能是好使的，也就是正常的通过性测试，按照接口文档上的参数，正常传入，是否可以返回正确的结果。  
②、参数组合：现在有一个操作商品的接口，有个字段type，传1的时候代表修改商品，商品id、商品名称、价格有一个是必传的，type传2的时候是删除商品，商品id　　是必传的，这样的，就要测参数组合了，type传1的时候，只传商品名称能不能修改成功，id、名称、价格都传的时候能不能修改成功。

③、接口安全：  
     1、绕过验证，比如说购买了一个商品，它的价格是300元，那我在提交订单时候，我把这个商品的价格改成3元，后端有没有做验证，更狠点，我把钱改成-3，是不是我的余额还要增加？  
     2、绕过身份授权，比如说修改商品信息接口，那必须得是卖家才能修改，那我传一个普通用户，能不能修改成功，我传一个其他的卖家能不能修改成功  
     3、参数是否加密，比如说我登陆的接口，用户名和密码是不是加密，如果不加密的话，别人拦截到你的请求，就能获取到你的信息了，加密规则是否容易破解。  
     4、密码安全规则，密码的复杂程度校验

④、**异常验证**：  
　　所谓异常验证，也就是我不按照你接口文档上的要求输入参数，来验证接口对异常情况的校验。比如说必填的参数不填，输入整数类型的，传入字符串类型，长度是10的，传11，总之就是你说怎么来，我就不怎么来，其实也就这三种，必传非必传、参数类型、入参长度。

2）、**根据业务逻辑来设计用例**  
　　根据业务逻辑来设计的话，就是根据自己系统的业务来设计用例，这个每个公司的业务不一样，就得具体的看自己公司的业务了，其实这也和功能测试设计用例是一样的。  
    　　举个例子，拿bbs来说，bbs的需求是这样的：  
  　　  1、**登录失败5次，就需要等待15分钟之后再登录**  
  　　  2、新注册的用户需要过了实习期才能发帖  
   　　 3、删除帖子扣除积分  
   　　 4、......  
   　　像这样的你就要把这些测试点列出来，然后再去造数据测试对应的测试点。

* Flowchart-4

**接口测试的流程**

* 需求评审，熟悉业务和需求
* 开发提供接口文档
* 编写接口测试用例
* 提测后开始测试
* 提交测试报告

1. API doc

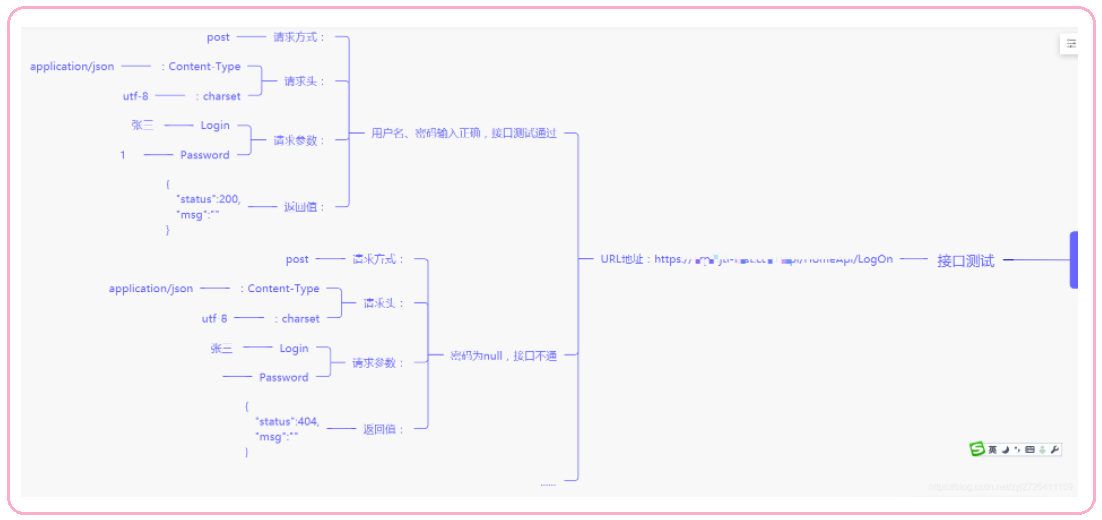
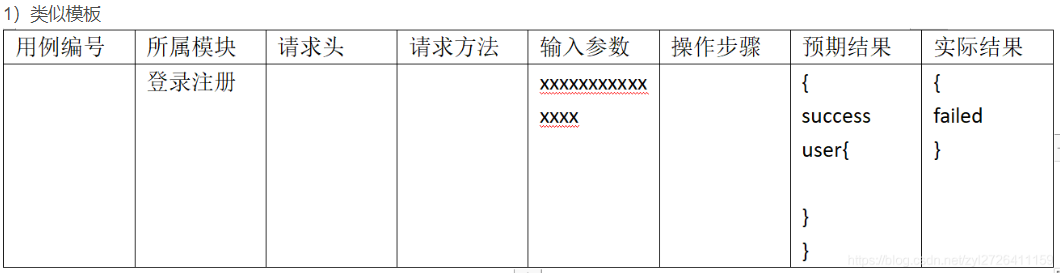
* Content
* API description,

Endpoint, request methods,

Request parameters and type,

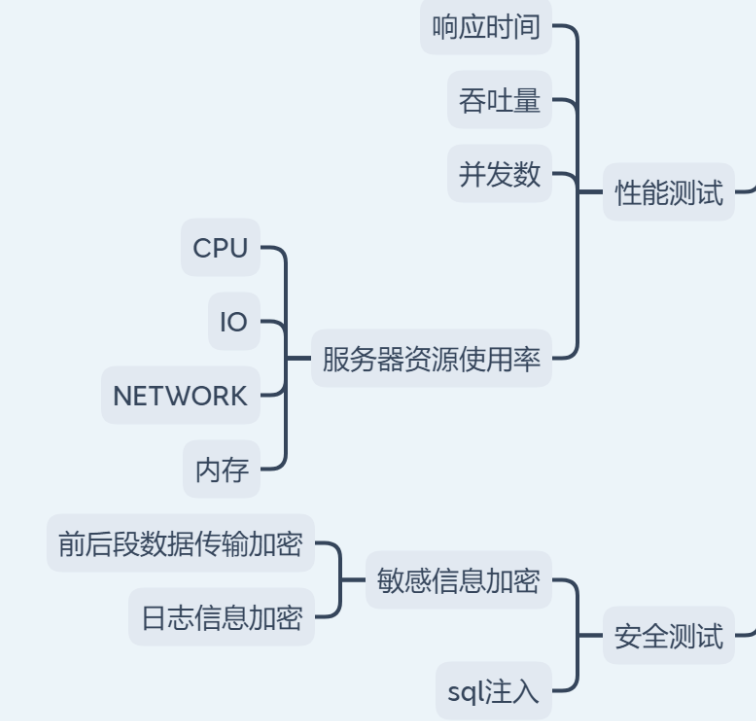
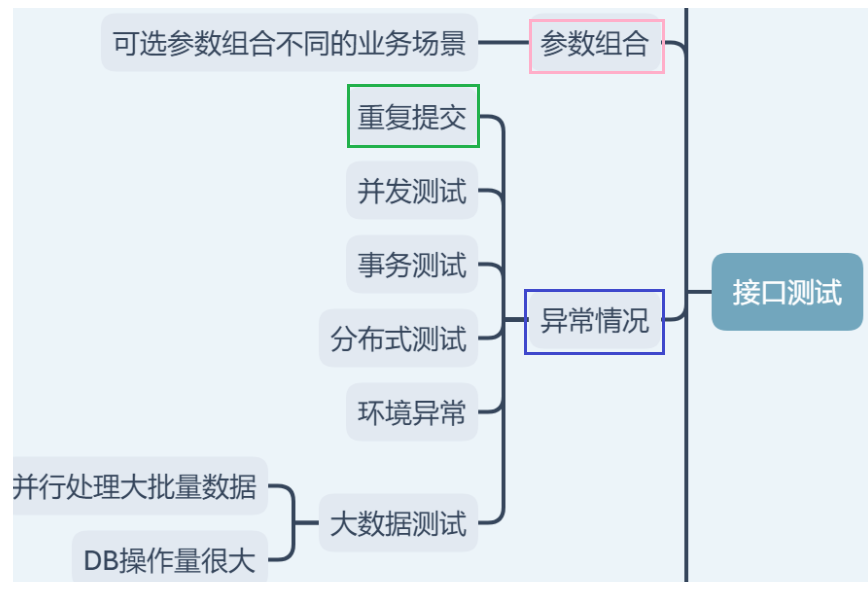
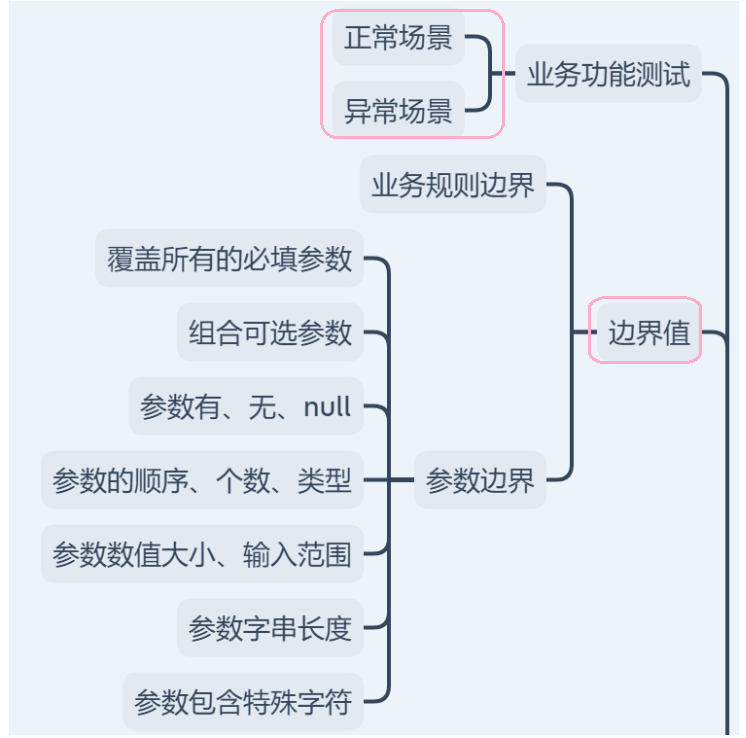
Payload

Return parameters and payload



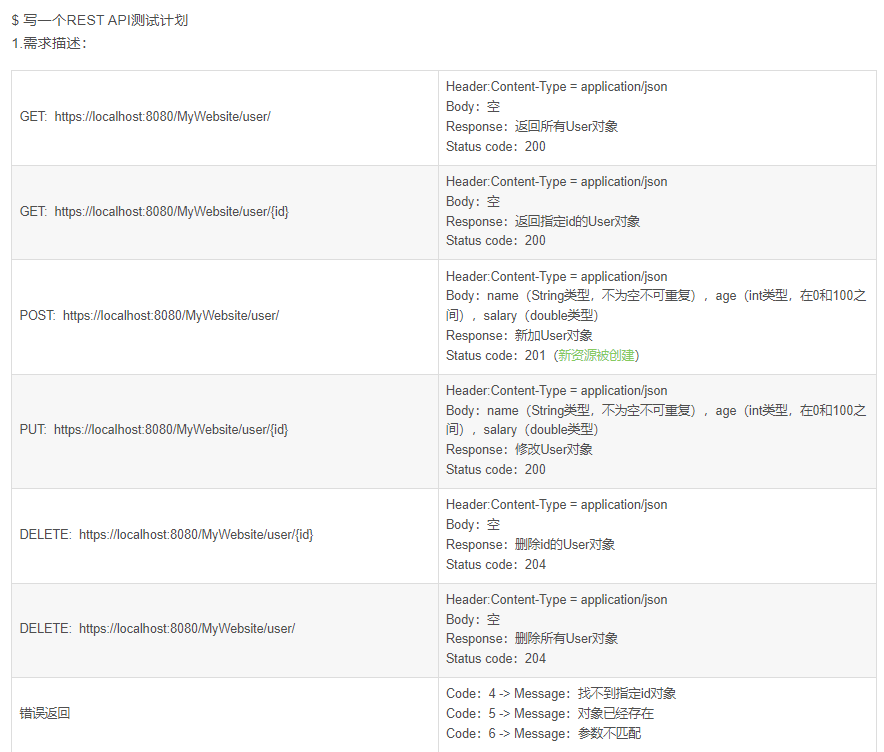
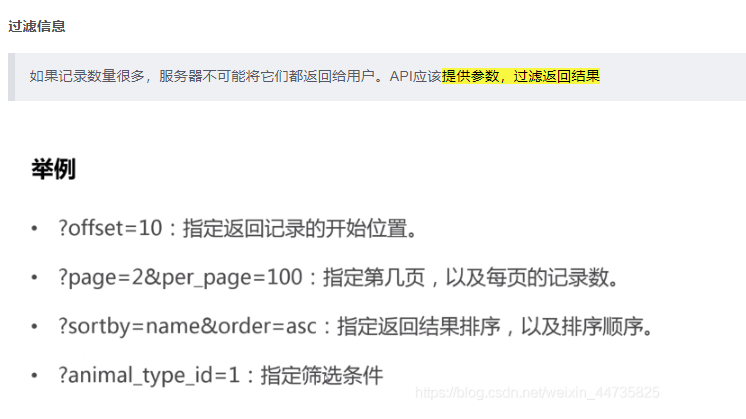
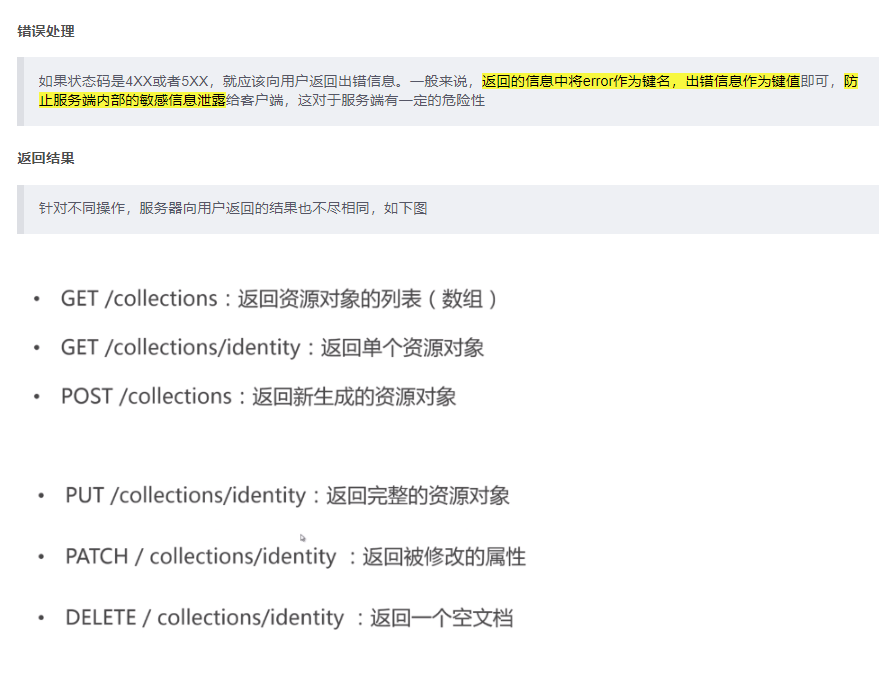
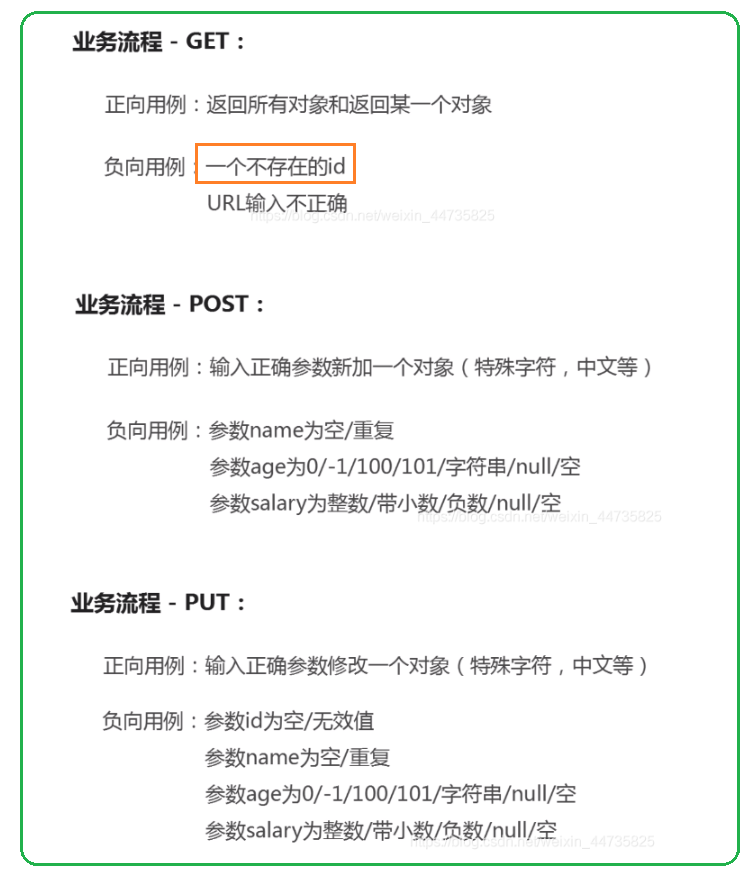
1. What to test

* Functionality
* If it works as expected.
* Business logic
* Exceptions
* Parameters
  + Keywords, num of keywords, types, or missing parameters
* Data
* Length, NO data, incorrect data



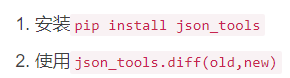
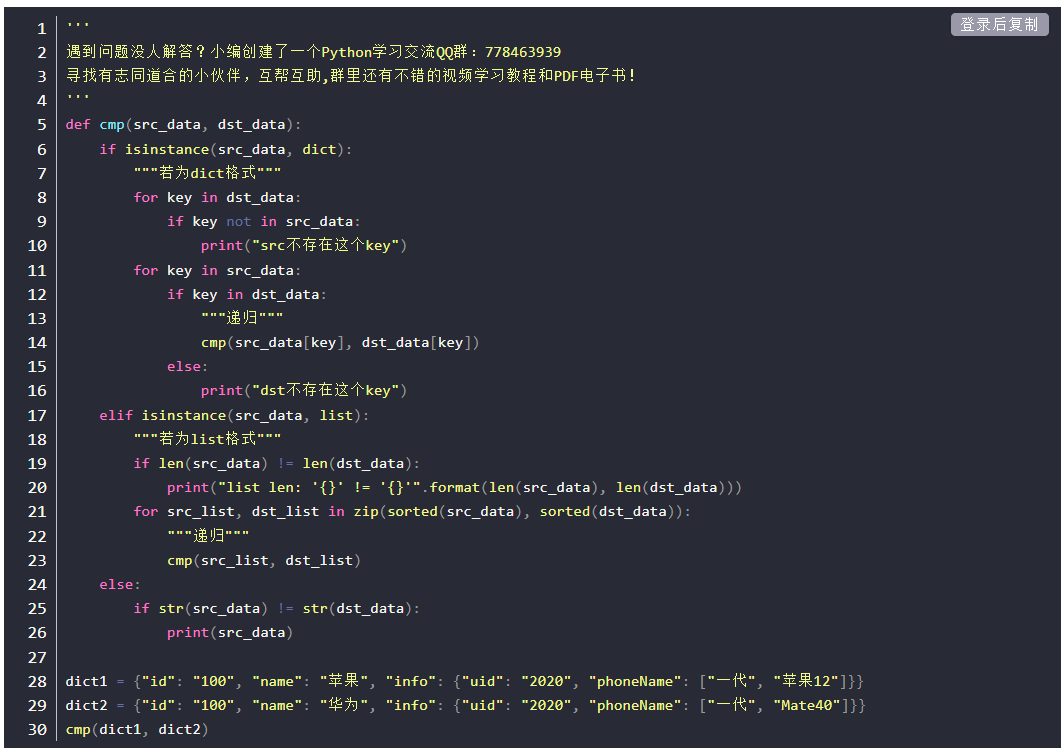
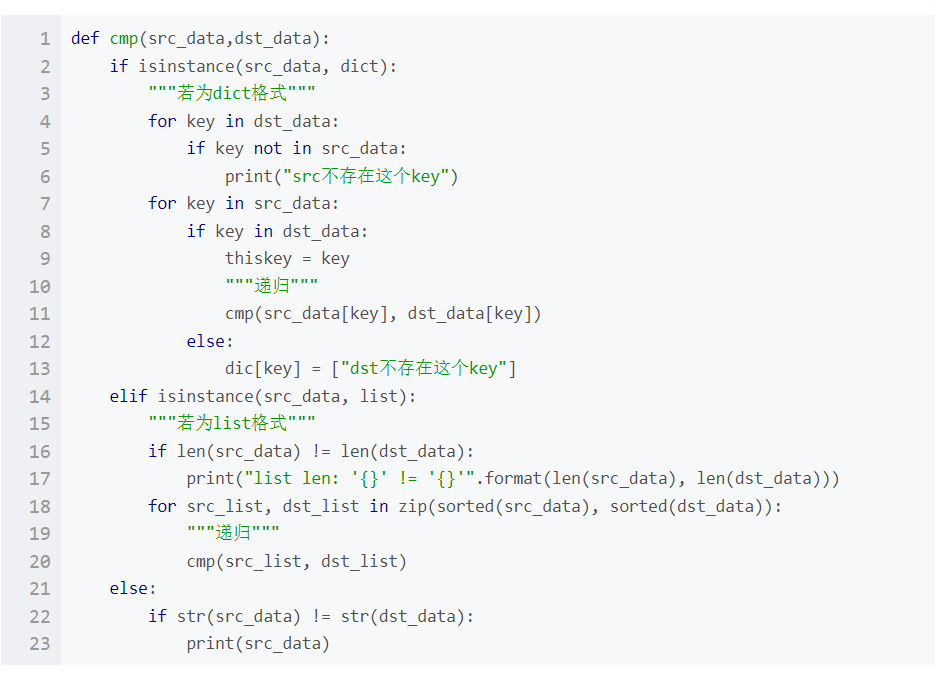
$ 性能测试  
测试覆盖：  
并发数  
吞吐量，tps  
出错率...

$ 安全性测试  
测试覆盖：  
测试覆盖敏感数据加密  
恶意攻击

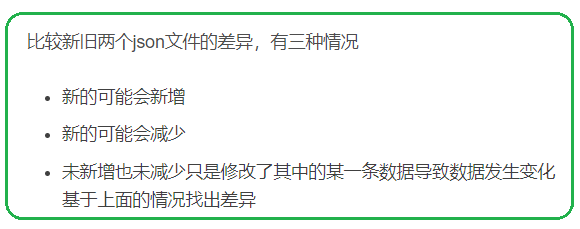
     

1. Json

* pip install jsonpath, from jsonpath import jsonpath
* <https://goessner.net/articles/JsonPath/>
* <https://blog.csdn.net/liuchunming033/article/details/106272542/?utm_medium=distribute.pc_relevant.none-task-blog-2~default~baidujs_baidulandingword~default-1.pc_relevant_paycolumn_v3&spm=1001.2101.3001.4242.2&utm_relevant_index=4>
* Compare two json files

   <https://blog.csdn.net/cjs68/article/details/81510805?spm=1001.2101.3001.6661.1&utm_medium=distribute.pc_relevant_t0.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1.pc_relevant_default&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1.pc_relevant_default&utm_relevant_index=1>

<https://blog.csdn.net/weixin_45005677/article/details/121036929?spm=1001.2101.3001.6650.3&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-3.pc_relevant_default&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-3.pc_relevant_default&utm_relevant_index=6>

 [https://blog.csdn.net/weixin\_47906106/article/details/121750319?ops\_request\_misc=&request\_id=&biz\_id=102&utm\_term=python%20%E6%AF%94%E8%BE%83json%20%E6%96%87%E4%BB%B6&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduweb~default-5-121750319.142^v9^pc\_search\_result\_control\_group,157^v4^control&spm=1018.2226.3001.4187](https://blog.csdn.net/weixin_47906106/article/details/121750319?ops_request_misc=&request_id=&biz_id=102&utm_term=python%20%E6%AF%94%E8%BE%83json%20%E6%96%87%E4%BB%B6&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-5-121750319.142%5ev9%5epc_search_result_control_group,157%5ev4%5econtrol&spm=1018.2226.3001.4187)

**print(result .elapsed.microseconds)# 获取微妙**

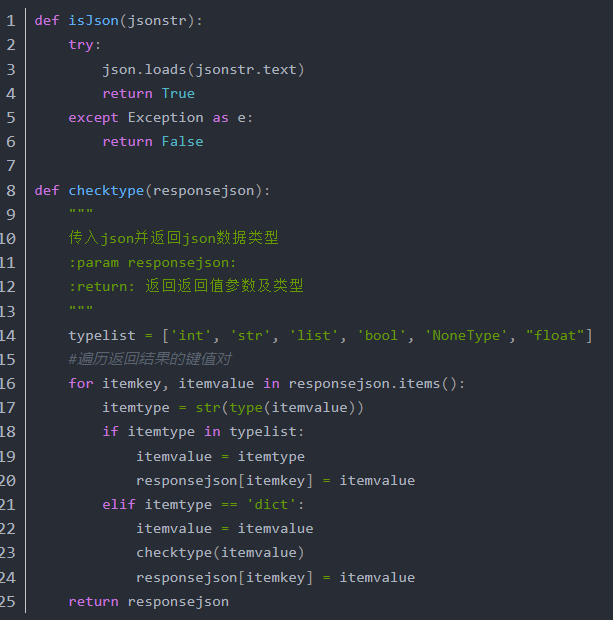
**print(result .elapsed.total\_seconds())# 获取秒**

# JSON Schema

## DeepDiff

# <https://blog.csdn.net/m0_56736369/article/details/117031305?spm=1001.2101.3001.6650.1&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1.pc_relevant_default&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1.pc_relevant_default&utm_relevant_index=2>

# pip install demjson, python 处理非标准 json 格式字符串

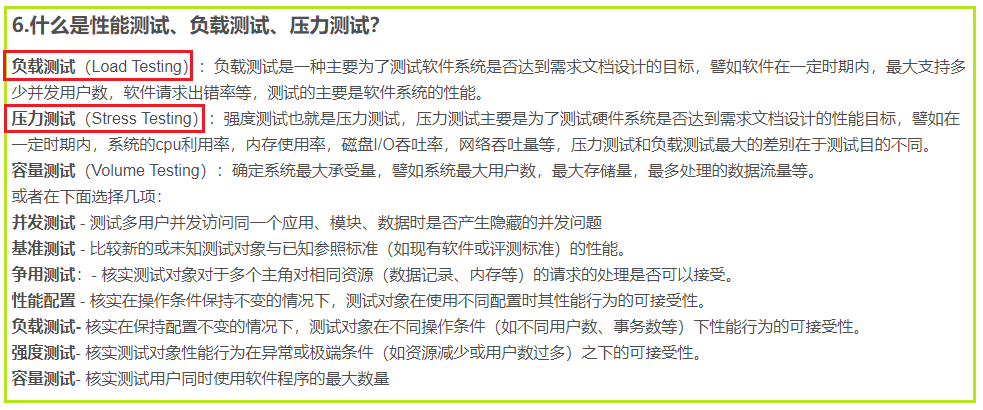
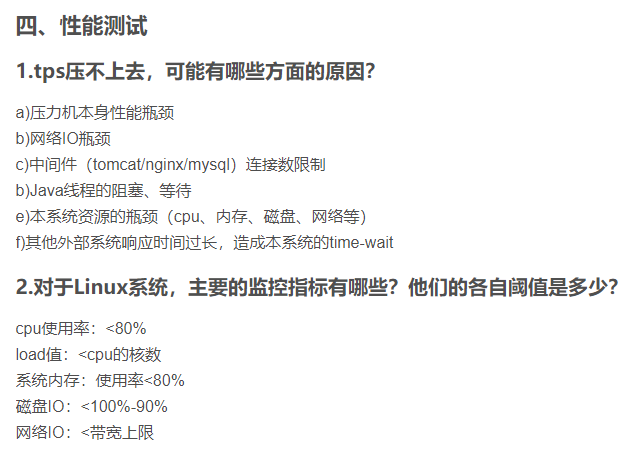
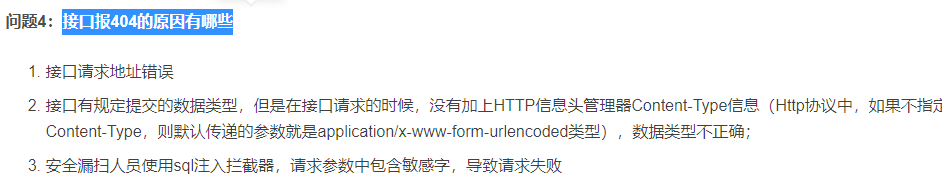


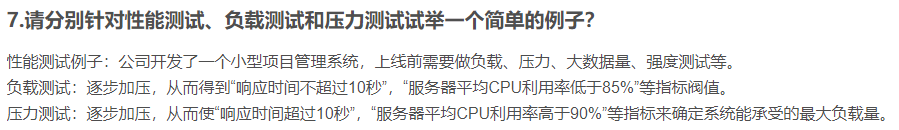
# 

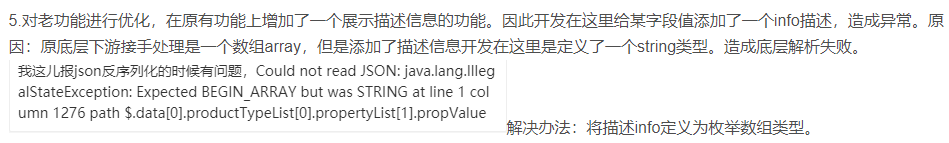
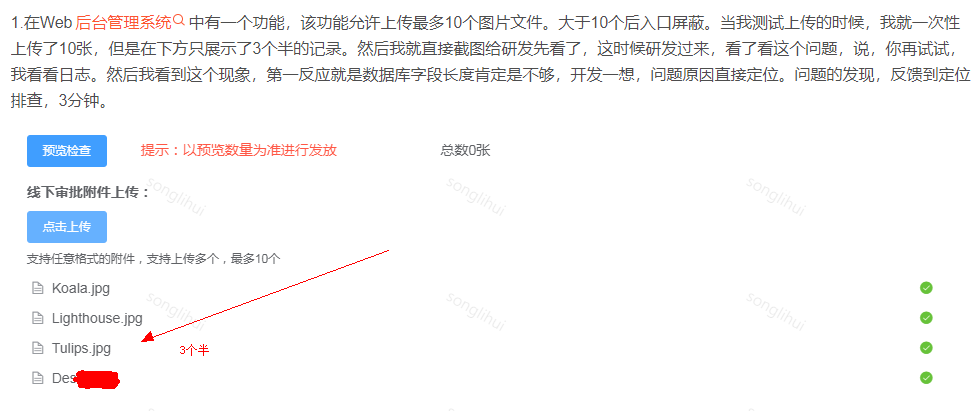
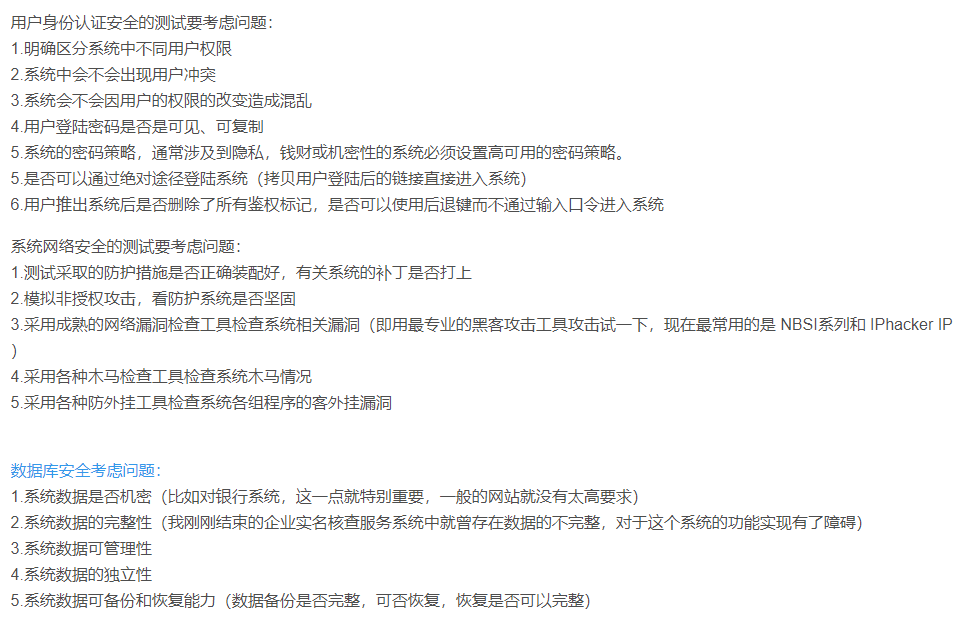
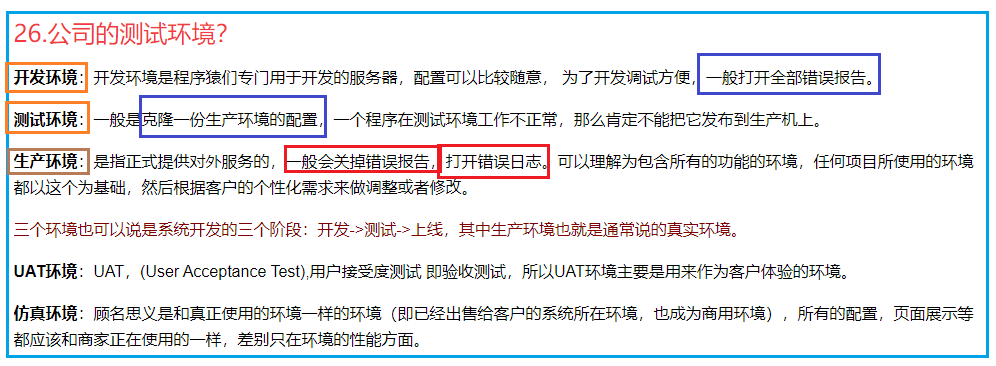
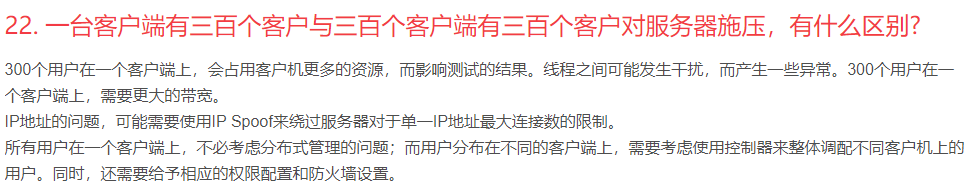
# Python操作数据库

# <https://blog.csdn.net/weixin_30438813/article/details/99780462?spm=1001.2101.3001.6650.2&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-2.pc_relevant_default&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-2.pc_relevant_default&utm_relevant_index=5>

# ss





11、一个登录界面如何涉及测试用例？

**界面UI测试**

1.布局是否合理，输入框，按钮对齐方式

2.输入框和按钮的高度，长度是否符合要求

3.界面的设计风格是否与UI的设计风格统一

4.界面的文字简洁易懂，没有错别字

**功能测试**

1.用户名，密码输入为空，点击查看提示信息

2.输入正确的用户名和密码，点击验证登录成功

3.登录成功后，验证是否跳转到正确的页面

4.用户名，密码如果过长，过短，是否有提示

5.用户名和密码前后有空格的处理

6.用户名和密码中有特殊字符或其他非英文的情况

7.记住用户名的功能

8.登录失败后，不能记住密码的功能

9.密码是否加密显示

10.登录页面的注册，忘记密码，登出等用另一账号登录链接是否正确

11.输入密码时，大写键盘开启是是否有提示信息

12.输入错误的用户名和密码，查看提示信息

**性能测试**

1.打开登录页面，需要几秒

2.输入正确的用户名和密码，登录成功不超过5s

**兼容性测试**

1.主流浏览器是否显示成功（IE8,9,10,11，Firefox，Chrome，Safafi）

2.不同的平台是否能显示成功（Mac，Windows)

3.移动设备上是否显示成功（Android，IOS）

4.不同的分辨率

**可用性测试**

1.是否支持全键盘操作，是否有快捷键

2.输入用户名和密码，按回车，是否可以登录

3.输入框能否可以Tab键切换

**安全测试**

1.登录成功后生成的Cookie，是否是Http only

2.用户名和密码是否通过加密的方式发给Web服务器

3.用户名和密码的验证，应该是在服务器端，而不是在Javascript前端

4.用户名和密码的输入框，应该屏蔽SQL注入

5.用户名和密码的输入框，应该禁止输入脚本

6.错误登录的次数限制

7.考虑是否支持多用户在同一机器上登录

8.考虑一用户在多台机器上登录