Table of Contents

接口测试课程

基于代码-项目实战

目标

1．熟悉接口自动化测试的流程
2．能够对一个项目的接口实现自动化测试

接口自动化测试流程

目标

1. 熟悉接口自动化测试的流程

---

1. 接口自动化测试的流程

1. 需求分析
2. 挑选需要做自动化测试的功能
3. 设计测试用例
4. 搭建自动化测试环境[可选]
5. 设计自动化测试项目的架构[可选]
6. 编写代码
7. 执行测试用例
8. 生成测试报告并分析结果

项目接口介绍

1. 项目介绍

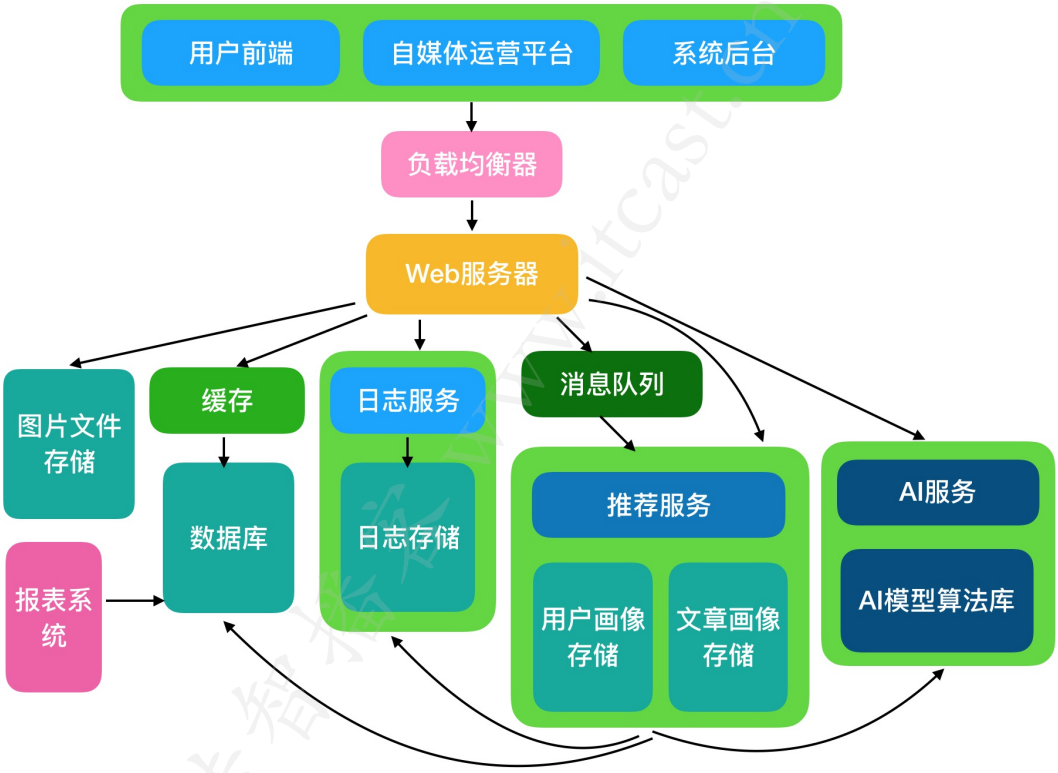
项目名称

黑马头条

项目描述

黑马头条是一款基于数据挖掘的推荐引擎产品，它为用户推荐有价值的、个性化的科技资讯，提供连接人与信息的新型服务。

项目技术架构



2. 项目接口分析

分析接口文档...

3. 挑选需要做接口测试的功能

...

用例设计

目标

1. 掌握如何编写自动化测试用例文档

1. 单接口的用例设计

ID	模块	接口名称	请求URL	用例名称	前置条件	请求类型	请求参数类型	请求参数	预期结果	测试结果	备注
001	登录	登录	http://ttapi.research.itcast.cn/app/v1_0/authorizations	请求参数为空		POST	json	{}	status_code: 400 返回数据：{"message": {"mobile": "Missing required parameter in the JSON body"}}		
002				手机号为空		POST	json	{"code": "888888"}	status_code: 400 返回数据：{"message": {"mobile": "Missing required parameter in the JSON body"}}		
003				短信验证码为空		POST	json	{"mobile": "13012345678"}	status_code: 400 返回数据：{"message": {"code": "Missing required parameter in the JSON body"}}		
004				手机号格式不正确		POST	json	{"mobile": "123456", "code": "888888"}	status_code: 400 返回数据：{"message": {"mobile": "123456 is not a valid mobile"}}		
005				短信验证码格式不正确		POST	json	{"mobile": "13012345678", "code": "abc"}	status_code: 400 返回数据：{"message": {"code": "Invalid params."}}		
006				短信验证码错误	成功获取短信验证码	POST	json	{"mobile": "13012345678", "code": "111111"}	status_code: 400 返回数据：{"message": "Invalid code."}		
007				短信验证码失效	成功获取短信验证码	POST	json	{"mobile": "13012345678", "code": "888888"}	status_code: 400 返回数据：{"message": "Invalid code."}		
008				登录成功	成功获取短信验证码	POST	json	{"mobile": "13012345678", "code": "888888"}	status_code: 201 返回数据: {"message":"OK","data":{"token":"x"}}		

2. 业务功能的用例设计

ID	模块	用例名称	接口名称	请求URL	请求类型	请求参数类型	请求参数	预期结果	测试结果
001	登录	登录成功	获取短信验证码	http://ttapi.research.itcast.cn/app/v1_0/sms/codes/:mobile	GET	url	手机号：13012345678	成功 返回数据：{"message":"OK"}	
			登录	http://ttapi.research.itcast.cn/app/v1_0/authorizations	POST	json	"mobile": "13012345678", "code": "8888"	登录成功 返回数据： {"message":"OK","data":{"token":"x"}}	
002	用户	用户频道列表	获取用户频道列表	http://ttapi.research.itcast.cn/app/v1_0/user/channels	GET			成功 返回数据： {"message":"OK","data":{"channels":[{"id":x, "name":"xxx"}]}}	
003	新闻	频道新闻推荐	获取频道新闻推荐	http://ttapi.research.itcast.cn/app/v1_0/articles	GET	QueryString	channel_id=x	成功 返回数据： {"message":"OK","data":{"page":1, "per_page":10, "results":[{"art_id":x, "title":"xxx"}]}}	
004	新闻	收藏文章	收藏文章	http://ttapi.research.itcast.cn/app/v1_0/user/collections	POST	json	{"target":1}	收藏成功 返回数据：{"message":"OK"}	
005	新闻	取消收藏文章	取消收藏文章	http://ttapi.research.itcast.cn/app/v1_0/user/collections/:target	DELETE	url	文章id：1	取消收藏成功 返回数据：{"message":"OK"}	

项目搭建

目标

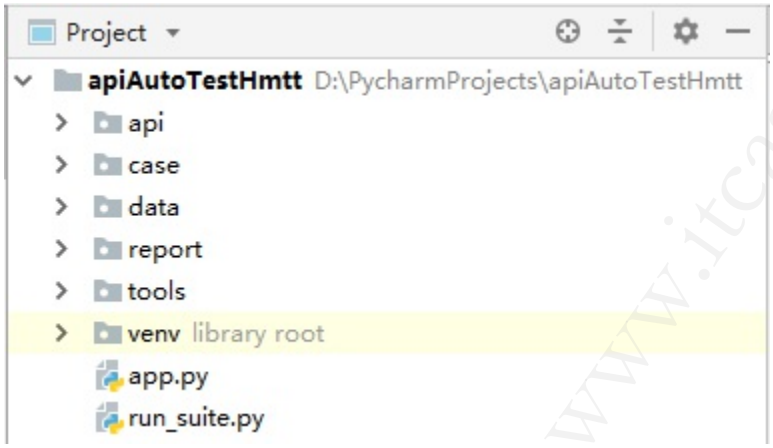
1. 掌握如何进行自动化测试框架的搭建

---

1. 新建项目

项目名称：apiAutoTestHmtt

2. 创建目录结构



3. 安装依赖包

```
pip install requests
```

编写代码

目标

1. 掌握如何封装项目中的接口
2. 掌握如何使用UnitTest管理项目中的测试用例

---

1. 封装接口类

根据用例分析待测功能，按功能模块定义接口类

```
登录：login.py
频道：channel.py
文章：article.py
收藏：collections.py
```

2. 编写测试脚本

1. 定义测试脚本文件

   ```
   登录模块：test_login.py
   频道模块：test_channel.py
   文章模块：test_article.py
   收藏模块：test_collections.py
   ```

2. 使用unittest管理测试脚本

3. 执行测试脚本

1. 使用unittest执行测试脚本
2. 调试代码

4. 数据库数据校验

4.1 用例场景

调用收藏文章的接口后，校验数据库中是否插入了对应的收藏记录。

4.2 表结构

```
CREATE TABLE `news_collection` (
  `collection_id` bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '主键id',
  `user_id` bigint(20) unsigned NOT NULL COMMENT '用户ID',
  `article_id` bigint(20) unsigned NOT NULL COMMENT '文章ID',
  `create_time` datetime NOT NULL COMMENT '创建时间',
  `is_deleted` tinyint(1) NOT NULL DEFAULT '0' COMMENT '是否取消收藏, 0-未取消, 1-已取消'
,
  `update_time` datetime NOT NULL COMMENT '更新时间',
  PRIMARY KEY (`collection_id`),
  UNIQUE KEY `user_article` (`user_id`,`article_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='用户收藏表';
```

4.3 示例代码

```
# 收藏
def test_collections(self):
    article_id = 1  # 文章id
    user_id = 1  # 用户id

    # 收藏
    response = self.collections_api.collections(article_id)

    # 断言响应数据
    json_result = response.json()
    self.assertEqual("OK", json_result.get("message"))

    # 数据库数据校验
    conn = pymysql.connect("localhost", "root", "root", "hmtt")
    cursor = conn.cursor()
    sql = "select collection_id,is_deleted from news_collection where user_id=%s and ar
ticle_id=%s"
    cursor.execute(sql, (user_id, article_id))
    data = cursor.fetchone()
    cursor.close()
    conn.close()
    self.assertIsNotNone(data)
    self.assertEqual(0, data[1])  # 未取消
```

4.4 封装数据库操作工具类

为了减少代码的冗余，提高测试效率，可以对数据库的相关操作封装成工具类。

示例代码

```python
import pymysql

class DBUtil:
    _conn = None  # 数据库连接对象

    @classmethod
    def get_conn(cls):
        """获取数据库连接对象"""
        if cls._conn is None:
            cls._conn = pymysql.connect("localhost", "root", "root", "hmtt")
        return cls._conn

    @classmethod
    def close_conn(cls):
        """关闭数据库连接"""
        if cls._conn:
            cls._conn.close()
            cls._conn = None

    @classmethod
    def get_cursor(cls):
        """获取游标对象"""
        return cls.get_conn().cursor()

    @classmethod
    def close_cursor(cls, cursor):
        """关闭游标对象"""
        if cursor:
            cursor.close()

    @classmethod
    def get_one(cls, sql):
        """查询一条记录"""
        data = None
        cursor = None
        try:
            cursor = cls.get_cursor()
            cursor.execute(sql)
            data = cursor.fetchone()
        except Exception as e:
            print("get_one error: ", e)
        finally:
            cls.close_cursor(cursor)
            cls.close_conn()
        return data
```

数据驱动

目标

1. 掌握如何把数据驱动应用到项目中

---

1. 数据驱动

1.1 定义数据文件

1. 定义存放测试数据的目录，目录名称：data
2. 分模块定义数据文件

```
登录模块：login.json
频道模块：channel.json
文章模块：article.json
收藏模块：collections.json
```

3. 根据业务编写用例数据

1.2 测试数据参数化

修改测试脚本，使用parameterized实现参数化

生成测试报告

目标

1. 掌握如何使用UnitTest生成测试报告

---

1. 生成测试报告

使用HTMLTestRunner生成测试报告

```
report_file = "./report/report{}.html".format(time.strftime("%Y%m%d-%H%M%S"))
with open(report_file, "wb") as f:
    runner = HTMLTestRunner(f, title="黑马头条接口自动化测试报告", description="V1.0")
    runner.run(suite)
```

2. 分析测试结果

...