Postman——API测试工具

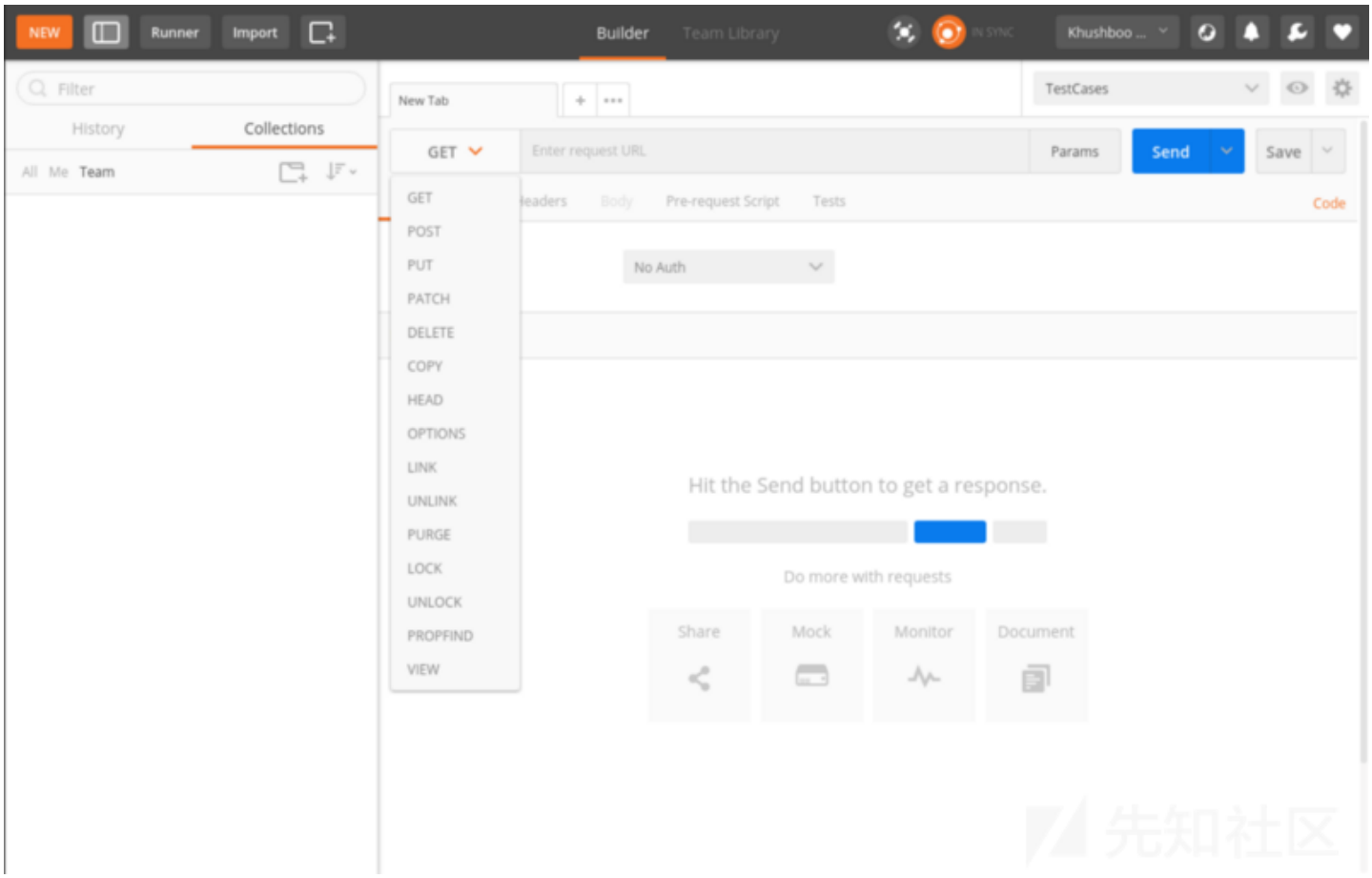
惊鸿一瞥最是珍贵 / 2019-01-22 13:27:00 / 浏览数 2609 安全工具 工具 顶(0) 踩(0)

摘要

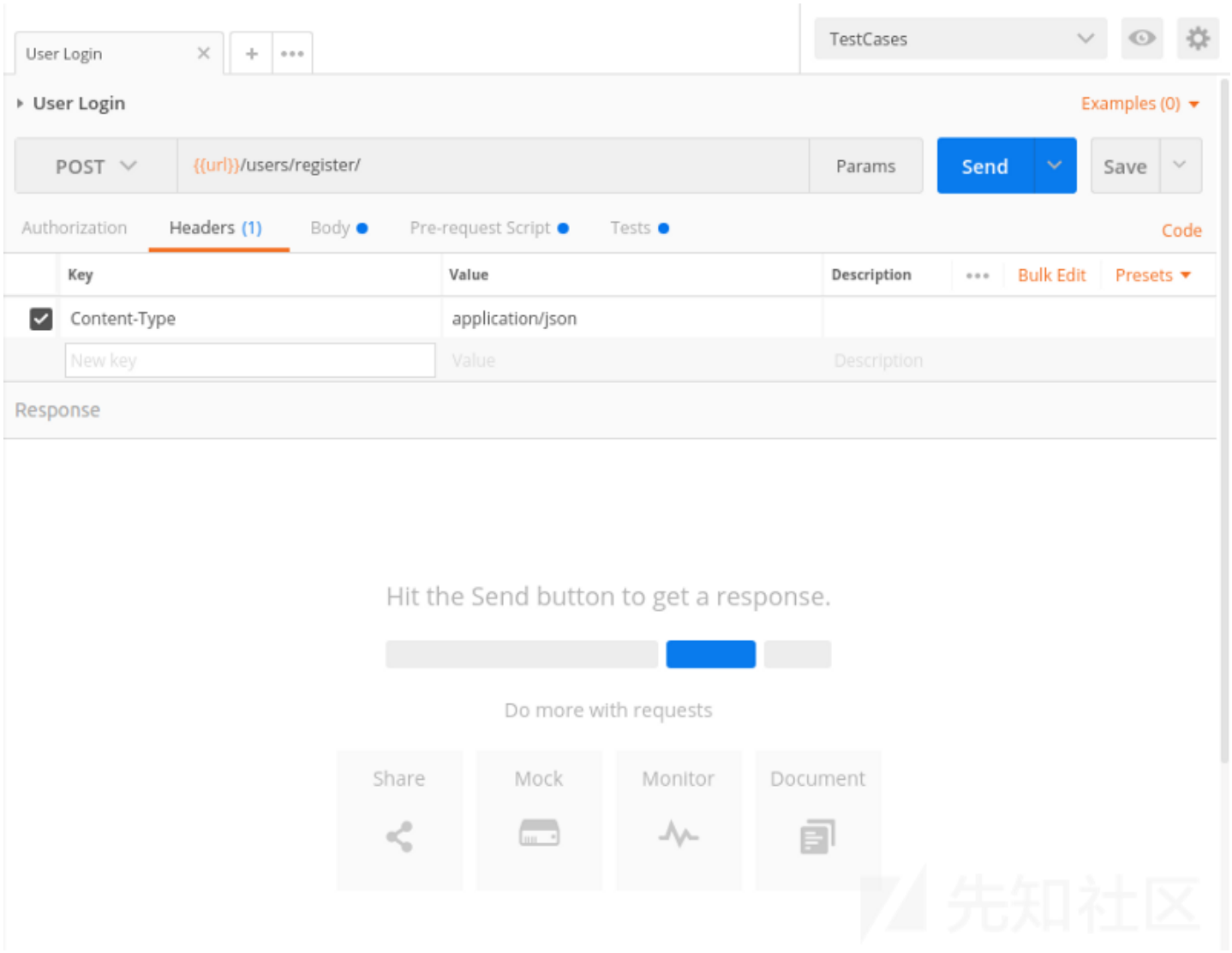
API代表应用程序编程接口。
API是用于构建应用程序软件的一组子程序定义，协议和工具。一般来说，这是一套明确定义的各种软件组件之间的通信方法。
API测试——测试API集合，检查它们的功能、性能、安全性，以及是否返回正确的响应。
API测试用于确定输出是否结构良好，是否对另一个应用程序有用，根据输入(请求)参数检查响应，并检查API检索和授权数据所花费的时间。
Postman是一个通过向Web服务器发送请求并获取响应来测试API的应用程序。

Postman安装

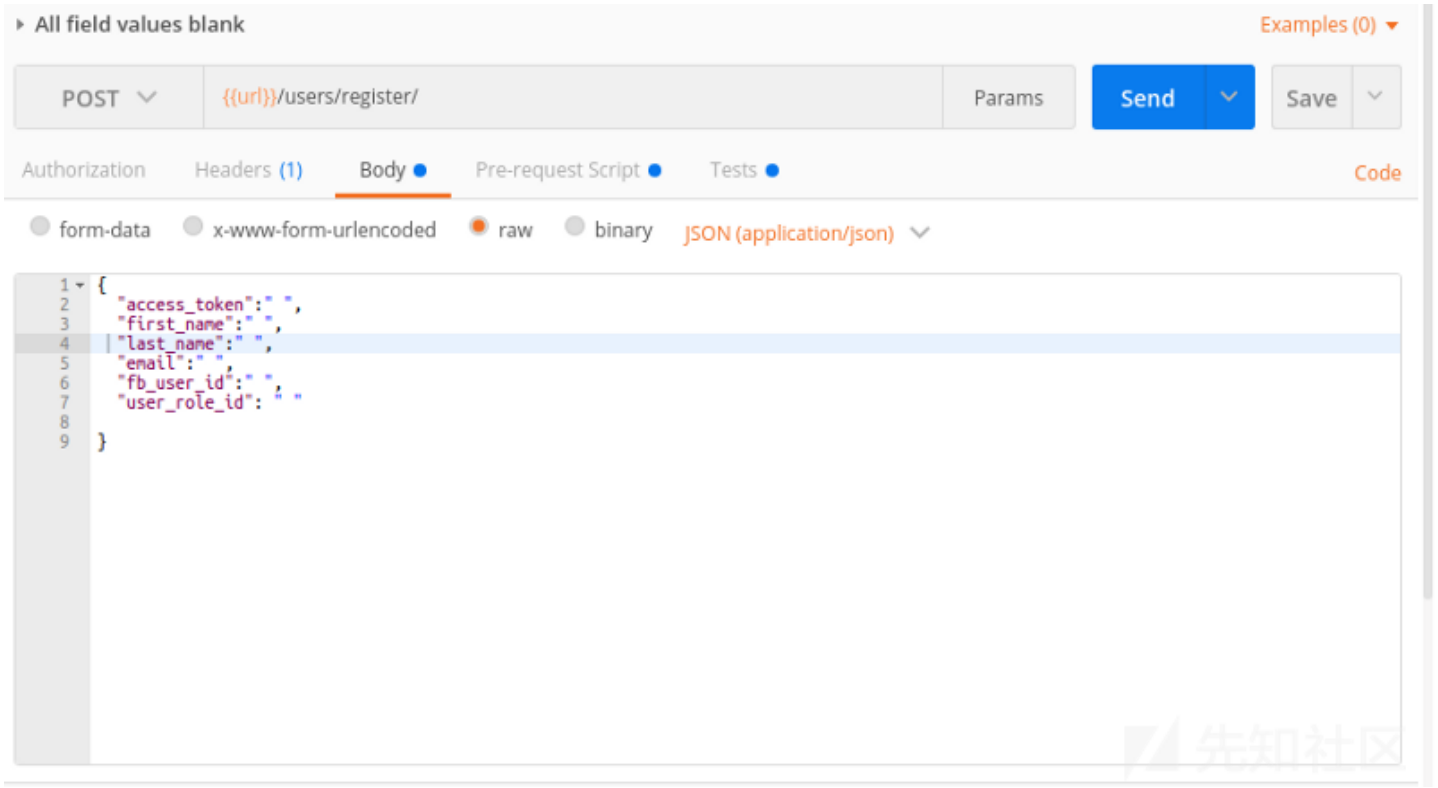
可以从以下URL下载Postman Native App：
https://www.getpostman.com/apps
或者你可以在Google
Chrome网上商店添加扩展程序，https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop?hl=en
Postman非常容易上手，它提供API调用的集合，我们必须按照规范来测试应用程序的API。
可以从给定的下拉列表中选择API调用方法，根据API调用设置授权、标头、正文等信息。
可在Postman中使用的API调用方法：



根据API调用的标头：



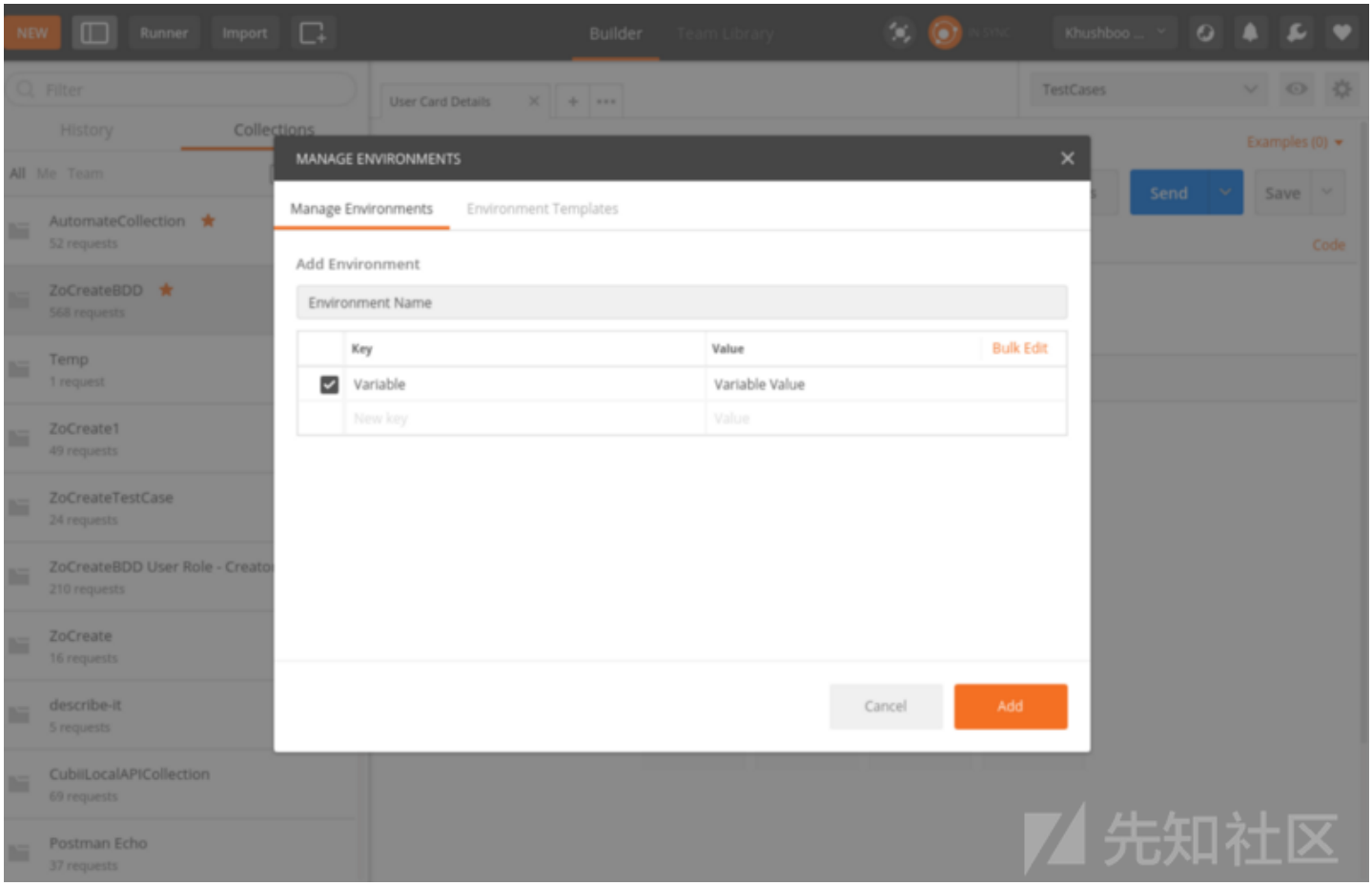
根据API调用的正文信息：



然后，您可以通过单击Send按钮来执行API调用。

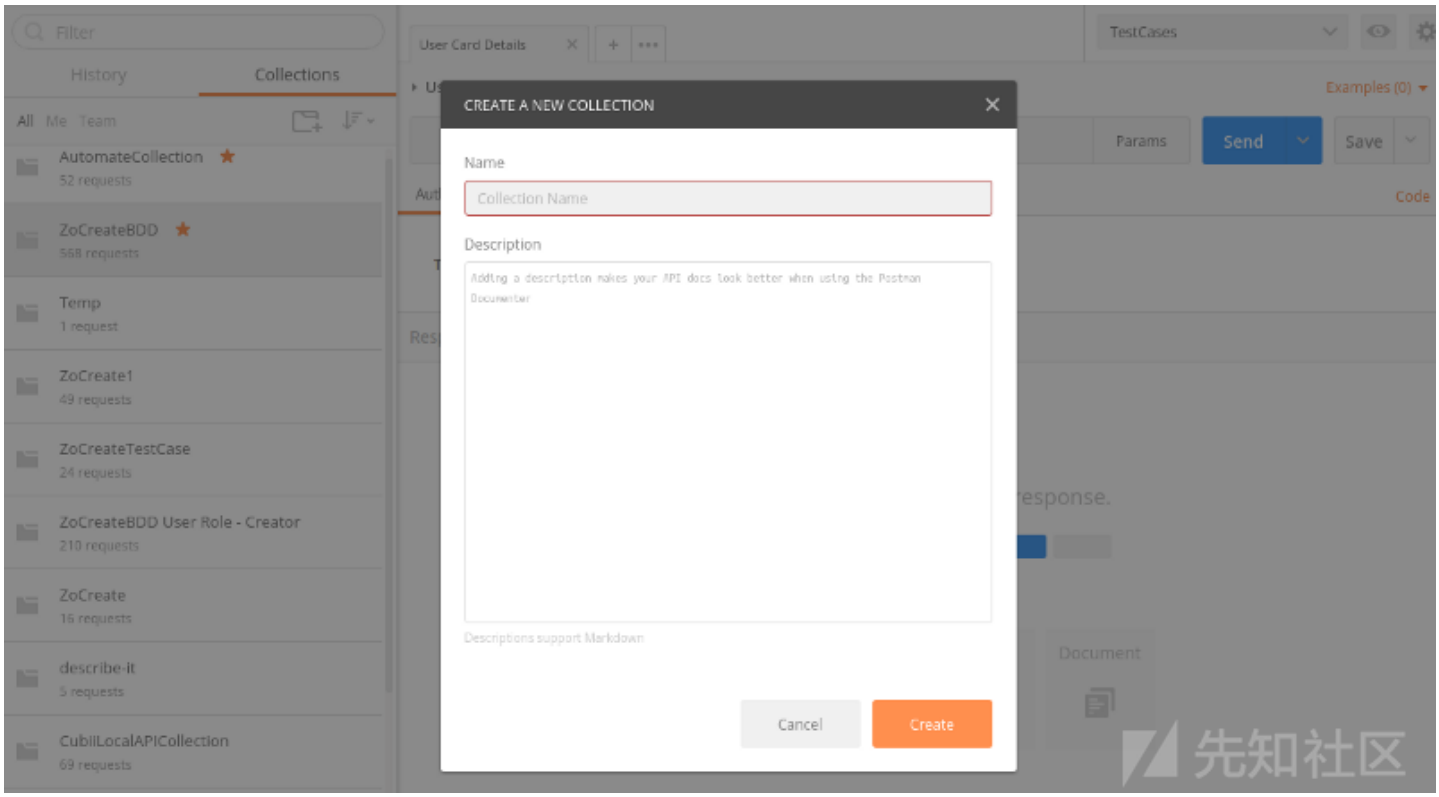
Postman中的环境变量

可以根据需要从右上角设置环境变量。可以通过以下步骤轻松设置环境变量：
1.单击“设置管理环境”(右上角的图标)。
2.单击“添加”按钮。
3.写下环境的名称。
4.填充键&值，以后可用作集合中的变量。

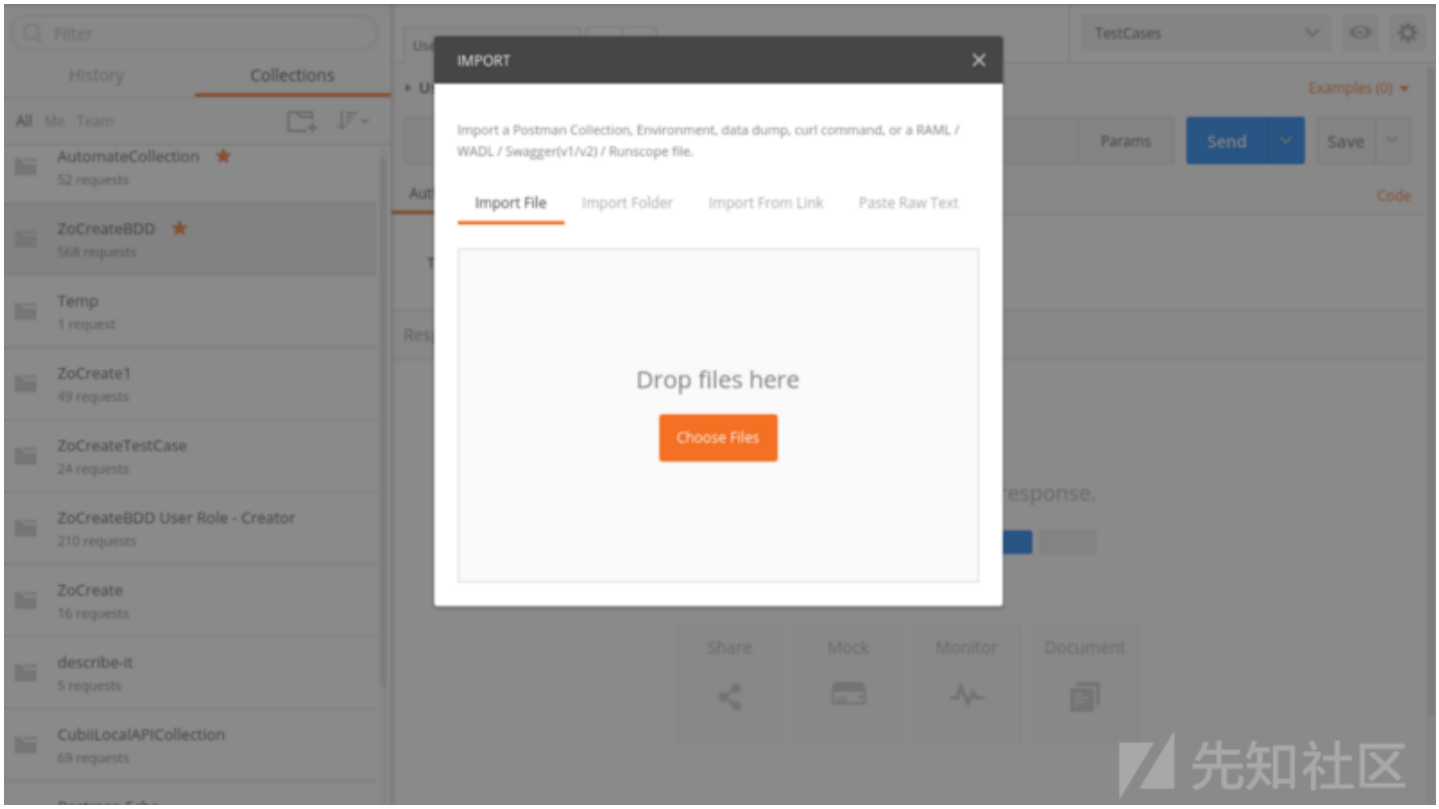


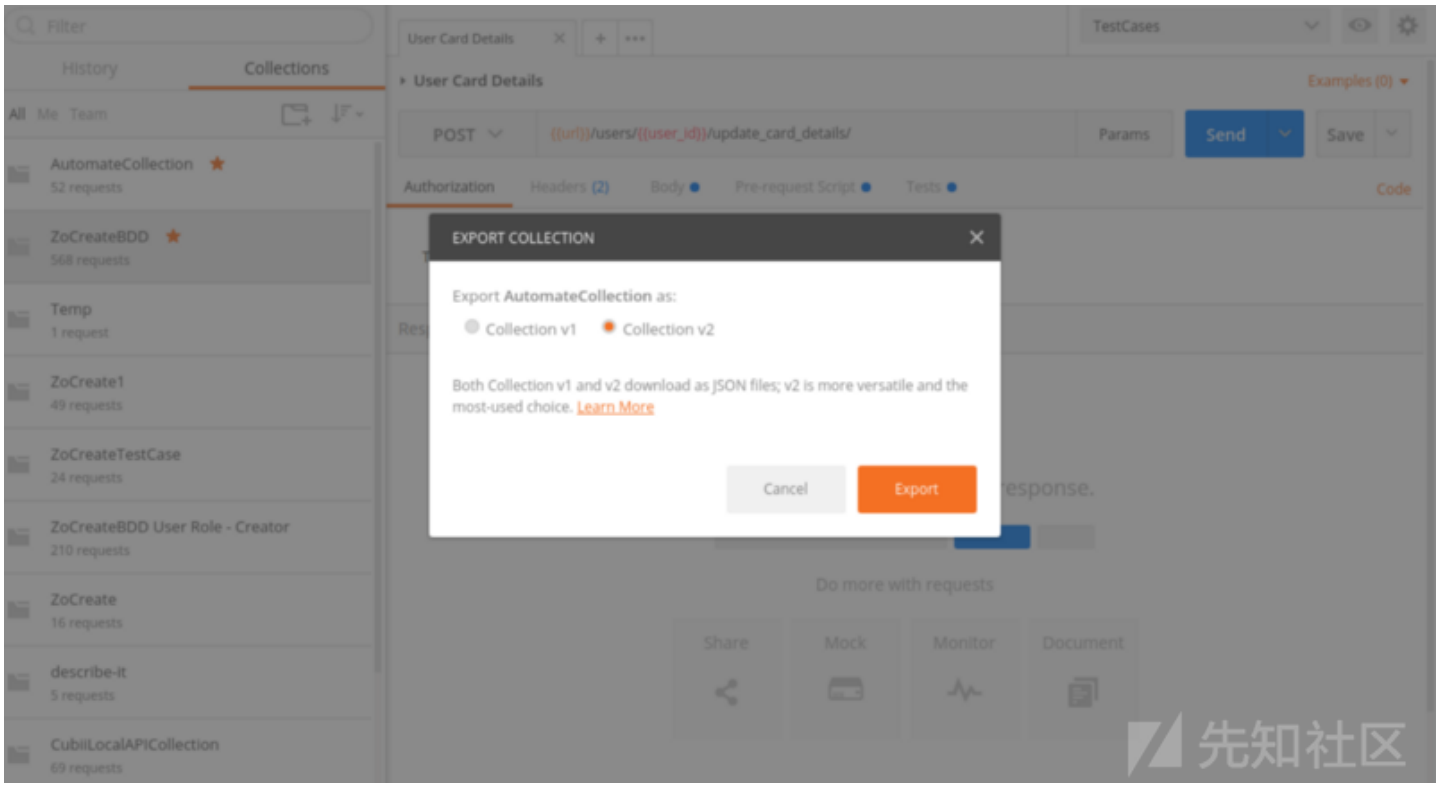
添加集合

您可以将每个API调用添加到集合中并创建一个集合，该集合可供应用程序重用。



一个人可以导入别人的集合，也可以导出他们的集合，这样其他人也可以在他们的电脑上使用这个集合。

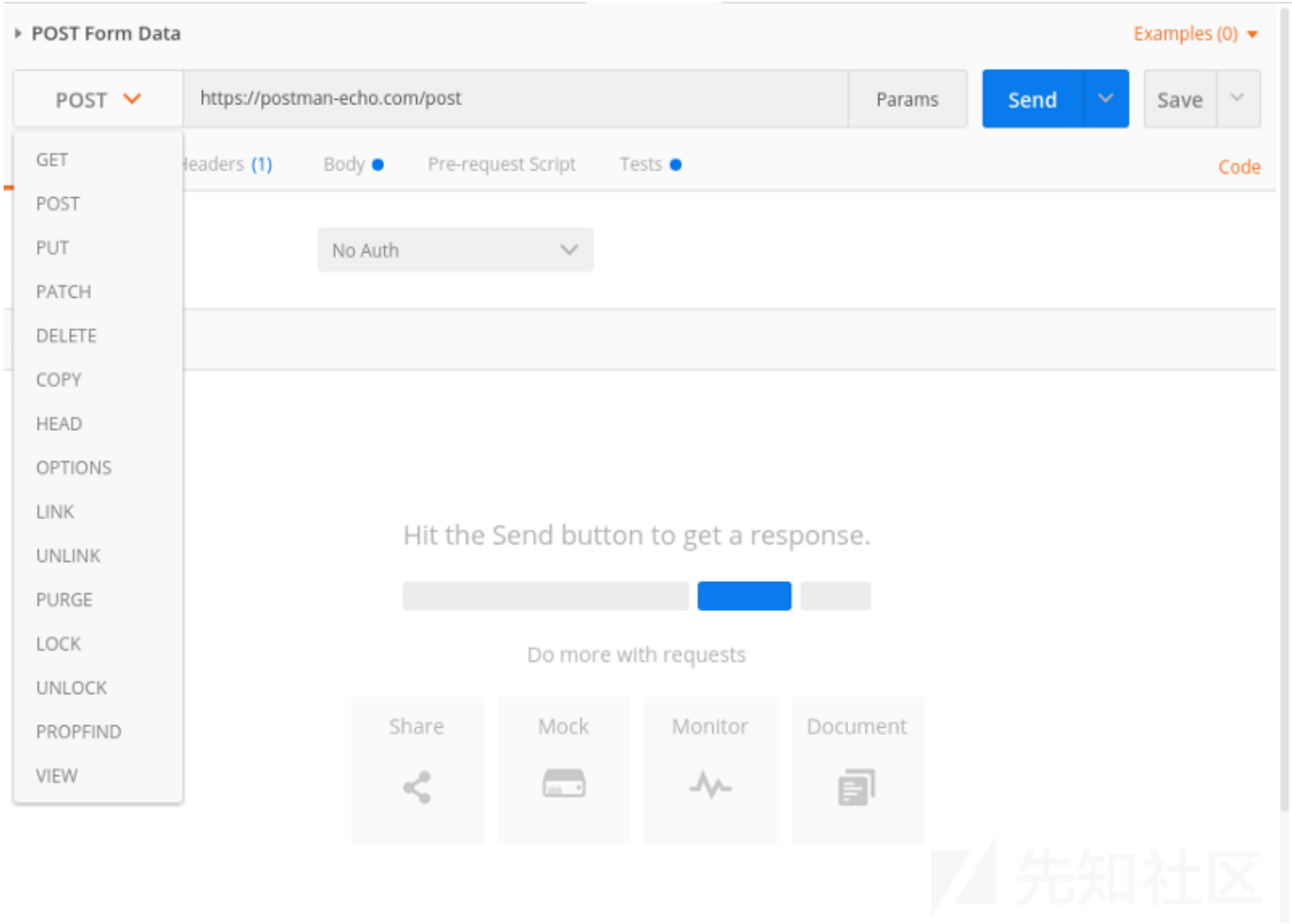




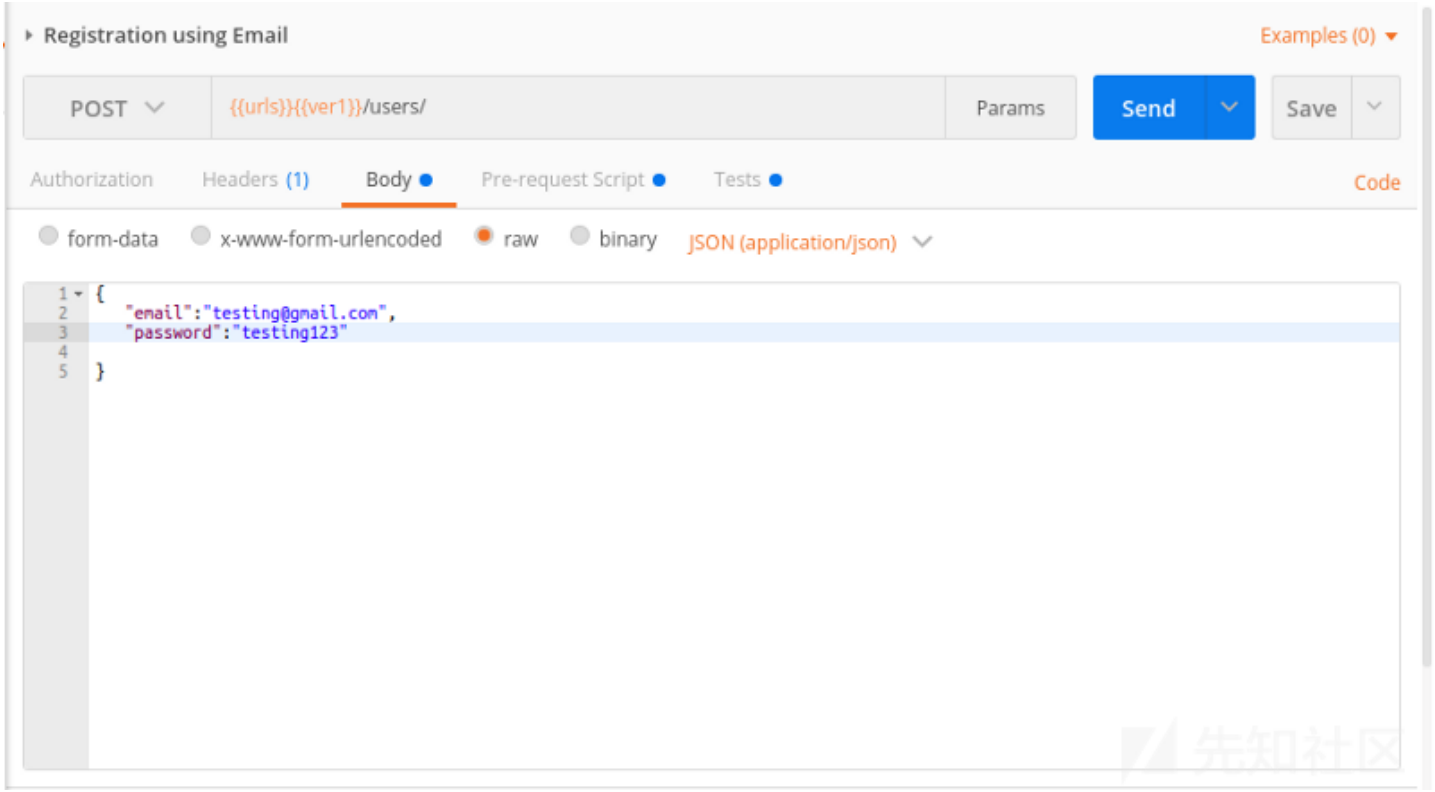
在API调用中，我主要使用了两种方法：

1.HTTP请求 - 请求是进行HTTP调用的最简单的方式。

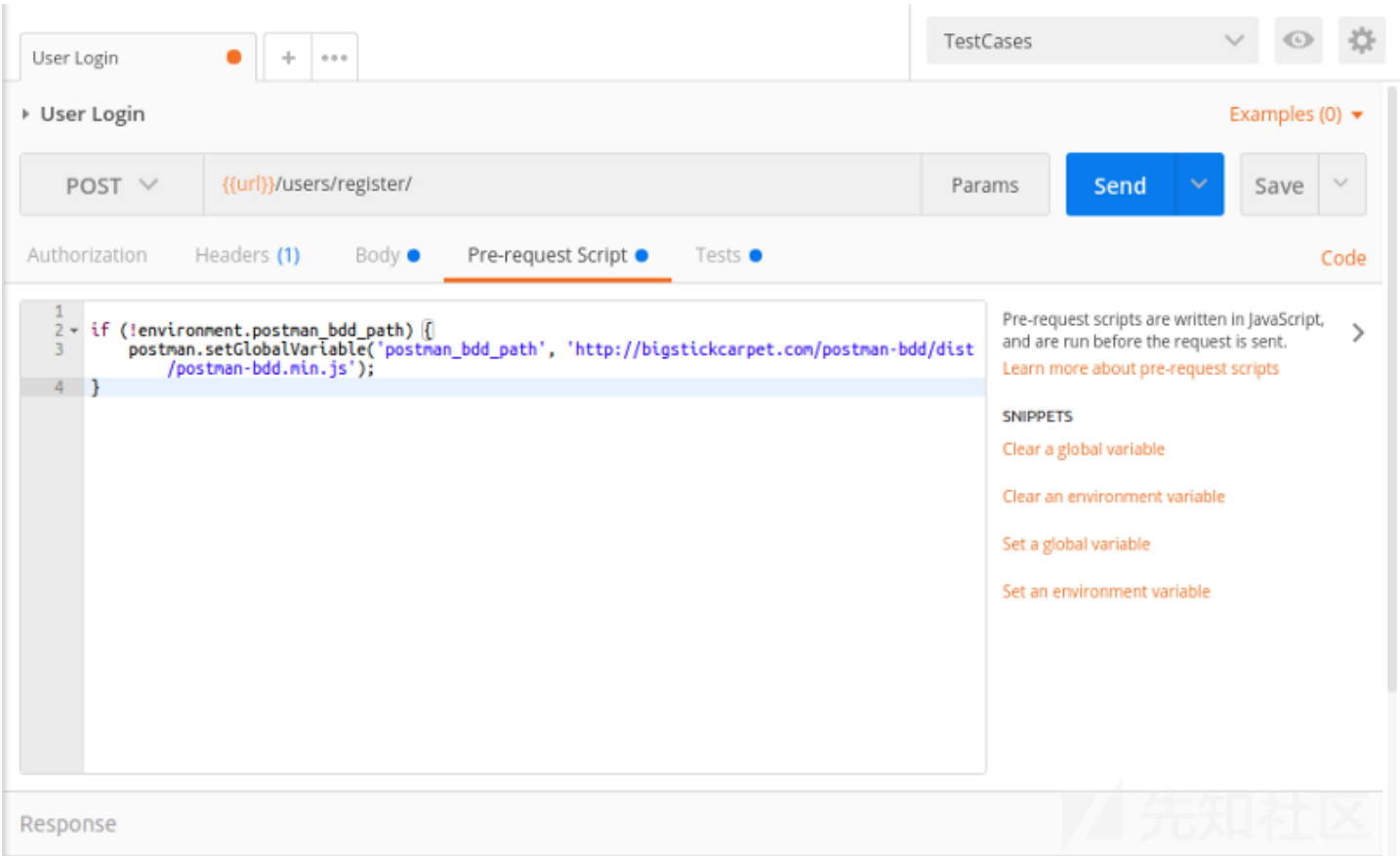
HTTP请求包含请求方法、请求URL、请求标头、请求主体、预请求脚本和测试（Request Method, Request URL, Request Headers, Request Body, Pre-request Script and Tests）。
请求方法（Request Method）-Request Methods定义要发出的请求类型。Postman中提供的请求方法如下所示：



有以下四种方法：
POST请求：创建或更新数据
PUT请求：更新数据
GET请求：用于检索/获取数据。
DELETE请求：用于删除数据
请求URL: 发出Http请求的位置
请求标头 - 在请求标头中它包含应用程序的键值。我主要使用了以下两个键值：
Content-Type - 内容类型描述对象数据的格式。内容类型，我在请求和响应中使用最多的是application/json。
Authorization - 请求中包含的授权令牌用于标识请求者。
请求主体（RequestBody）- 它包含要随请求一起发送的数据(取决于请求方法的类型)。我使用原始形式的数据发送请求。示例如下：



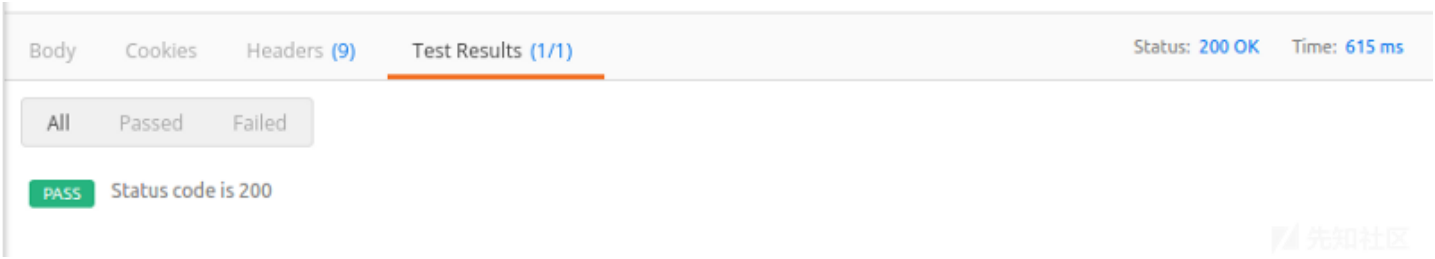
预请求脚本 - 预请求脚本是在发送请求之前执行的一段代码。
示例：为了在请求中使用PostmanBDD(本文后面将对此进行解释)，需要在预请求脚本中定义以下代码。



Postman的测试:在Postman中，可以使用JavaScript语言为每个请求编写和运行测试。以下是示例：

测试描述示例：

测试结果示例：



2.HTTP响应——在发送请求时，API发送响应，包括正文，Cookie，标头，测试，状态代码和API响应时间。
Postman在不同的选项卡中组织正文和标题。完成API调用所花费的时间的状态代码显示在另一个选项卡中。
有许多状态代码，我们可以从这些代码验证响应。

```
200 - ■■■■■■
201 - ■■■■■■■■■■■■
204 - ■■■■■
400 - ■■■■■■■■■■■■■■■■■■■■■■■■
401 - ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
403 - ■■■■■■■■■■■■
404 - ■■■■■■■
405 - ■■■■■■■■■■■■■■■■
500 - ■■■■■■■■■
503 - ■■■■■■■
```

Postman中的测试脚本

有了Postman，就可以使用JavaScript语言为每个请求编写和运行测试。收到响应后，将在“测试”选项卡下添加代码并执行。

```
tests[“Status code is 200”] = responseCode.code ===200;
```

将检查收到的响应代码是否为200。

您可以对一个请求进行任意多个测试。大多数测试都是简单的，只有一条线性JavaScript语句。下面是更多的例子。

检查响应主体是否包含字符串：

```
tests["Body matches string"] = responseBody.has("string_you_want_to_search");
```

检查响应主体是否等于特定字符串：

```
tests["Body is correct"] = responseBody === "response_body_string";
```

检查JSON值：

```
var data = JSON.parse(responseBody);
tests["Your test name"] = data.value === 100;
```

检查响应时间是否小于200毫秒：

```
tests["Response time is less than 200ms"] = responseTime < 200;
```

检查成功的POST请求状态代码：

```
tests["Successful POST request"] = responseCode.code === 201 || responseCode.code === 202;
```

检查响应标头类型：

```
tests[‘The Content-Type is JSON’] = postman.getResponseHeader(‘Content-Type’) === ‘application/json’;
```

Postman BDD

Postman BDD允许使用BDD语法来构造测试，使用Fluent CHAI-JS语法来编写断言。因此，上面的测试用例可以如下所示：

检查响应标题类型：

```
it(‘should return JSON’, () => {
response.should.be.json;
response.should.have.header(‘Content-Type’, ‘application/json’);
response.type.should.equal(‘application/json’);
});
```

检查状态代码为200：

```
it(‘should be a 200 response’, () => {
response.should.have.status(200);                                });
```

检查响应时间是否小于200毫秒：

```
it(‘should respond in a timely manner’, () => {
response.time.should.be.below(200);
});
```

检查响应正文消息应为“用户成功登录”：

```
it(‘message should contain’, () => {
response.body.message.should.equal(‘User logged in successfully.’) ;
});
```

Postman BDD的优点和缺点

简单的语法。
它具有简单的语法，使测试更易于编写和读取。
错误处理。
如果脚本中出现错误，则只有一个测试失败，而其他测试仍在运行，并显示错误。
丰富的断言。
它提供了对所有CHAI-JS和CHAI-HTTP断言以及API的一些自定义断言的完全访问权限。断言更容易记住和可读，例如自定义断言response.body.shand.be.a.user。
JSON模式验证。
用户可以使用Assertion作为response.body.should.have.schema(someJsonSchema)再次验证特定的JSON模式的响应

安装Postman BDD

安装Postman BDD有两个简单步骤：
1.下载
使用以下URL在Postman中创建GET请求：
http://bigstickcarpet.com/postman-bdd/dist/postman-bdd.min.js
2.安装
用户必须在按上述方式创建的请求中的“测试”选项卡中添加以下代码：

```
postman.setGlobalVariable('postmanBDD', responseBody);
```

然后，在全局安装Postman BDD。您可以在任何Postman请求中使用它：

```
eval(globals.postmanBDD);
```

总结

Postman对API测试非常有用，它会使您的任务变得更加简单高效。

链接：https://medium.com/aubergine-solutions/api-testing-using-postman-323670c89f6d

目录