**LAB 11: THỰC HÀNH VỚI WORKFLOW**

**Giới thiệu tổng quan**

Ở bài lab trước, chúng ta đã được làm quen với Collection runner khi thực thi một loạt request trong một collection từ đầu đến cuối mà không cần quan tâm đến thứ tự ưu tiên của chúng.

Tuy nhiên, trong một dự án, sẽ có những nghiệp vụ cụ thể mà chúng ta cần phải xây dựng các kịch bản test cũng như thứ tự thực thi các request theo một luồng nhất định.

Bài lab này chúng ta sẽ tập trung vào :

* Luyện tập kỹ năng tạo workflow/scenario test để thực thi tập request.
* Thực hành chạy tự động Collection với Collection runner.

**Kiến thức liên quan:**

* Basic Javascript.
* Kiến thức tổng quan về Trello, được giới thiệu ở bài lab số 8.
* DELETE là để xóa tài nguyên yêu cầu: Phương thức DELETE yêu cầu máy chủ gốc xóa tài nguyên được xác định bởi URI yêu cầu. Phương pháp này CÓ THỂ bị ghi đè bởi sự can thiệp của con người (hoặc các phương tiện khác) trên máy chủ gốc

**I/ Yêu cầu thực hành**

1. Thực hành viết script để kiểm tra nghiệp vụ và tính đúng đắn dữ liệu trả về của API
2. Thực hành chạy các request theo một trình tự hay một kịch bản nhất định, từ đó có thể hiểu hơn về luồng nghiệp vụ và xây dựng kịch bản test theo luồng nghiệp vụ.
3. Sử dụng Collection runner để chạy tập API.

**II/ Ví dụ mẫu**

**Đề bài.**

Cho danh sách API và một workflow như sau:

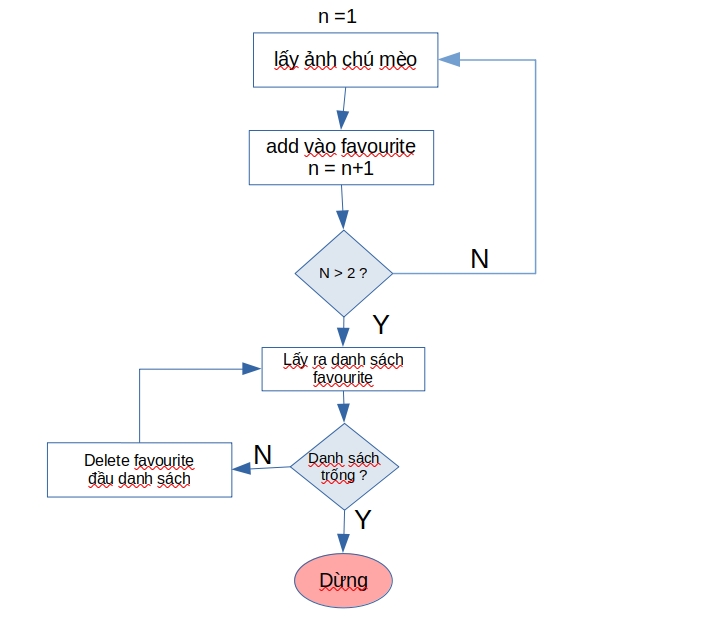
**API yêu cầu AUTHEN.**cần add **x-api-key** vào header. Tạm thời chúng ta sẽ sử dụng key sau để test:

**DEMO-API-KEY**

Danh sách API.

| **API** | **Param/response** | **Requirement** |
| --- | --- | --- |
| **GET:** https://api.thecatapi.com/v1/images/search?breed\_ids={breed-id}  Mô tả: API trả về ngẫu nhiên hình ảnh một chú mèo thuộc giống {breed-id} được cung cấp | **Path params:** breed-id: required. (id của giống mèo)  **response**  {  breeds :   (mảng các breed)  id : id ảnh  url : đường link download ảnh  width : độ chiều rộng của ảnh  height: chiều dài ảnh  } | - Nếu hành công, API cần trả về HTTP 200 OK  - Nếu Không thành công, API có thể trả về response code khác 200 .  - Nếu breed-id không tồn tại, APi trả về 404 not found nội dung response cần có thông tin sau : message : "Breed not found (master breed does not exist)" |
| **POST**  https://api.thecatapi.com/v1/favourites  Mô tả: API thêm một bức ảnh vào danh sách yêu thích theo account hiện tại | Body:  {  "image\_id": id image  "sub\_id": user account id  }  Response  {  "message": "SUCCESS",  "id": id của favourites  } | - Nếu thiếu image\_id, API cần trả về 400 Bad request cùng với trường message là  "\"image\_id\" is not allowed to be empty"  -Nếu nhập vào một image\_id không hợp lệ, cần trả về 404 Not found.  -Nếu thiếu trường sub\_id. API trả về 400 Bad request và message sẽ phải là  "\"sub\_id\" is not allowed to be empty"  -Nếu thành công thì trả về 200 Ok và message là “success” cùng với id của vote. |
| **GET**  https://api.thecatapi.com/v1/favourites  Mô tả: Lấy ra danh sách favourites theo user | **request param order:** optional(ASC|DESC sắp xếp tăng, giảm)  **limit:** optional giới hạn số lượng vote lấy về)  **sub\_id** : id user account  response: vd [  {  "id": 2145650,  "user\_id": "4",  "image\_id": "SCHe- SekW",  "sub\_id": "tester\_01",  "created\_at": "2022-01-27T16:27:24.000Z",  "image": {  "id": "SCHe-SekW",  "url": "https://cdn2.thecatapi.com/images/SCHe-SekW.jpg"  }  }, | -Nếu thành công thì trả về 200 Ok. và danh sách favourites của userId đó. Việc tạo favoutite phải cần có userId, do đó nếu :  -Nếu để userId trống, API cần trả về 400 Bad request với message là:  “User is not allowed to be empty"  - Nếu vào userId không tồn tại thì API trả về 404 Not Found |
| **Delete**  https://api.thecatapi.com/v1/favourites/{favourite\_id}  Mô tả: Xóa bỏ một favourite. | **request param** favourite\_id**:** id  response:  {  "message":  "SUCCESS| FAILURE"  } | - Nếu thành công thì trả về 200 Ok vùng message là “SUCCESS”. - Các trường hợp còn lại → xóa thất bại |

**#WorkFlow.**



**Yêu cầu:**

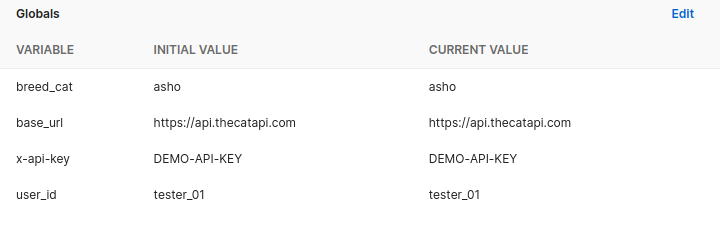
1. Đặt 1 user account bất kì (ở đây tôi lấy ví dụ là **tester\_01**)

**2.** Lần lượtlấy random ảnh của 2 chú mèo thuộc giống **“asho”** và add vào danh sách favourite theo account có id là “**tester\_01**”  
3. Lấy ra danh sách tất cả các favourite theo user : “**tester\_01**”  
4.Thực hiện xóa favourite ở đầu danh sách.

5. Lặp lại từ bước 2 cho đến khi không còn phần tử nào trong danh sách thì dừng lại.

**Thực hành.  
1. Thiết lập các biến môi trường.**

| **Key** | **Value** | **Type** | **Description** |
| --- | --- | --- | --- |
| base\_url | [https://api.thecatapi.com](https://api.thecatapi.com/) | global | thiết lập trước |
| x-api-key | DEMO-API-KEY | global | thiết lập trước |
| breed\_cat | asho | global | thiết lập trước |
| user\_id | tester\_01 | global | thiết lập trước |
| num\_image |  | global | thiết lập động, cập nhật sau mỗi lần get image |
| image\_id |  | global | thiết lập động, cập nhật sau mỗi lần get image |
| favourite\_id |  | global | thiết lập động, gán cho phần tử đầu tiên danh sách sau khi call api get all favourite |

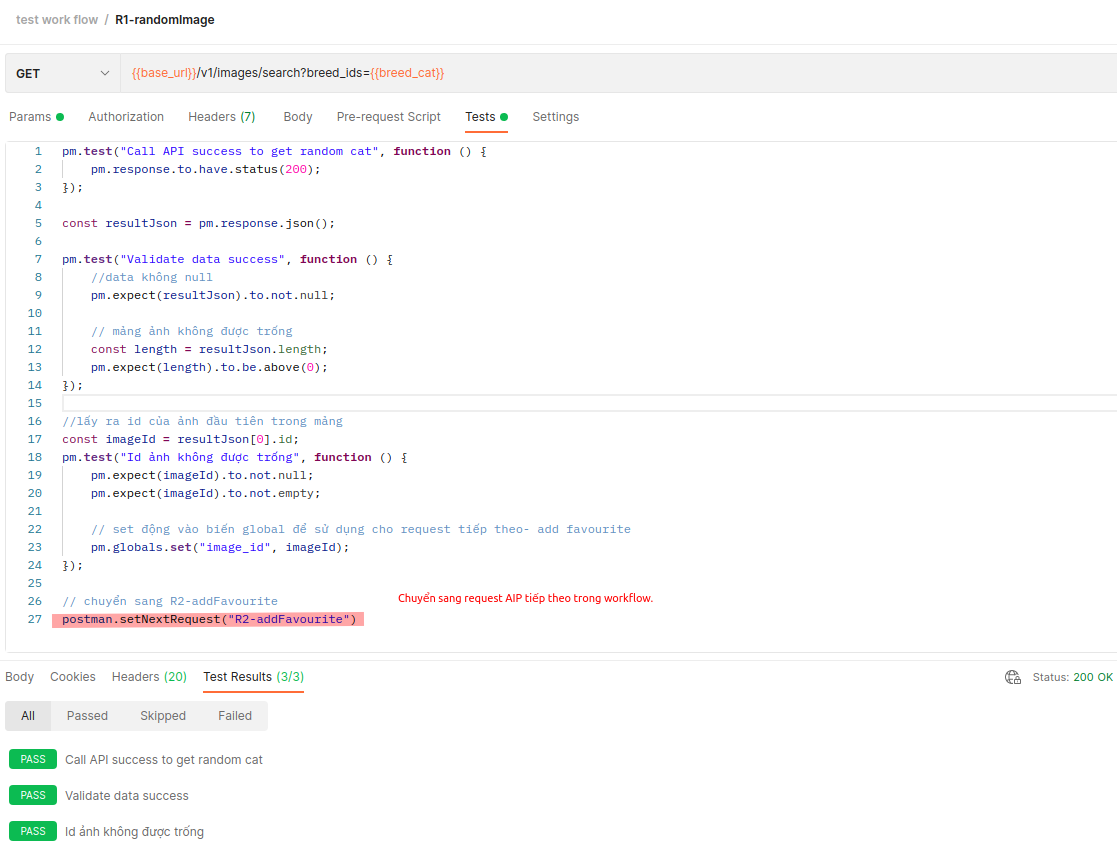


**2. Tạo các request và các testScript cho collection.**

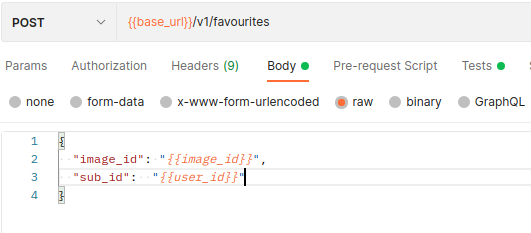
Add authen header cho tất cả các request.



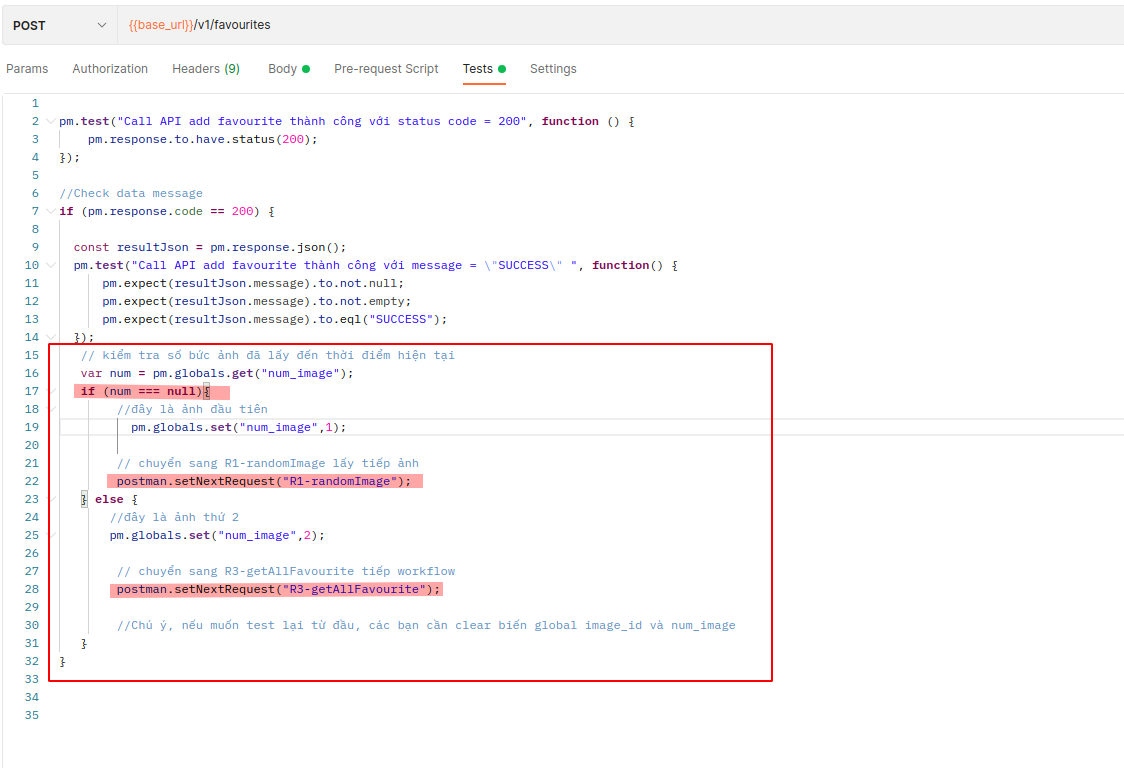
**2.1 R1-randomImage**



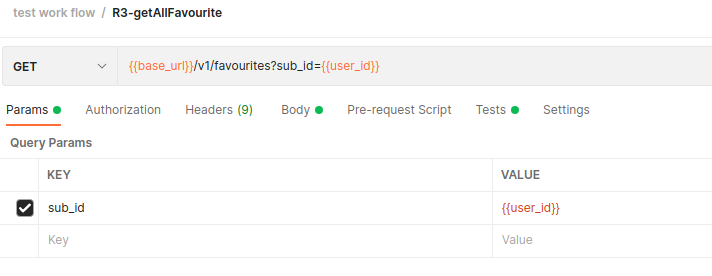
**2.2 R2-addFavourite**



#script



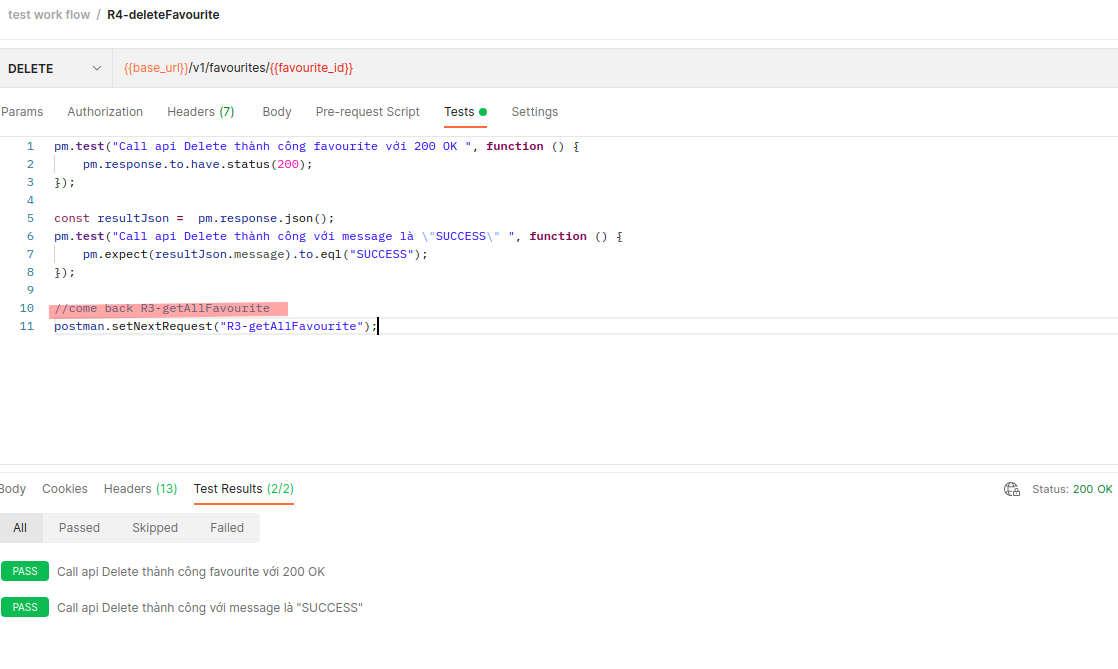
**2.3 R3-getAllFavourite**



Script.

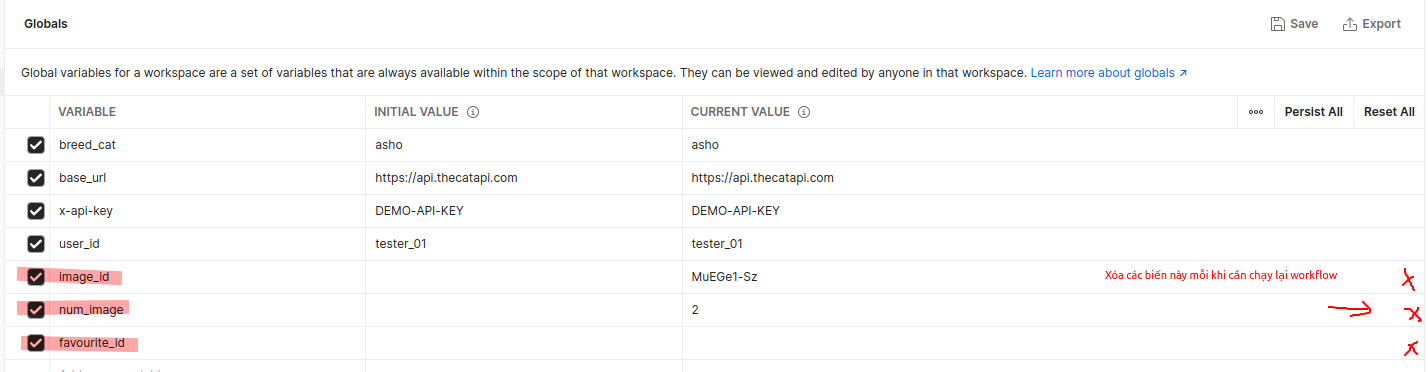


**2.4 R4-deleteFavourite**

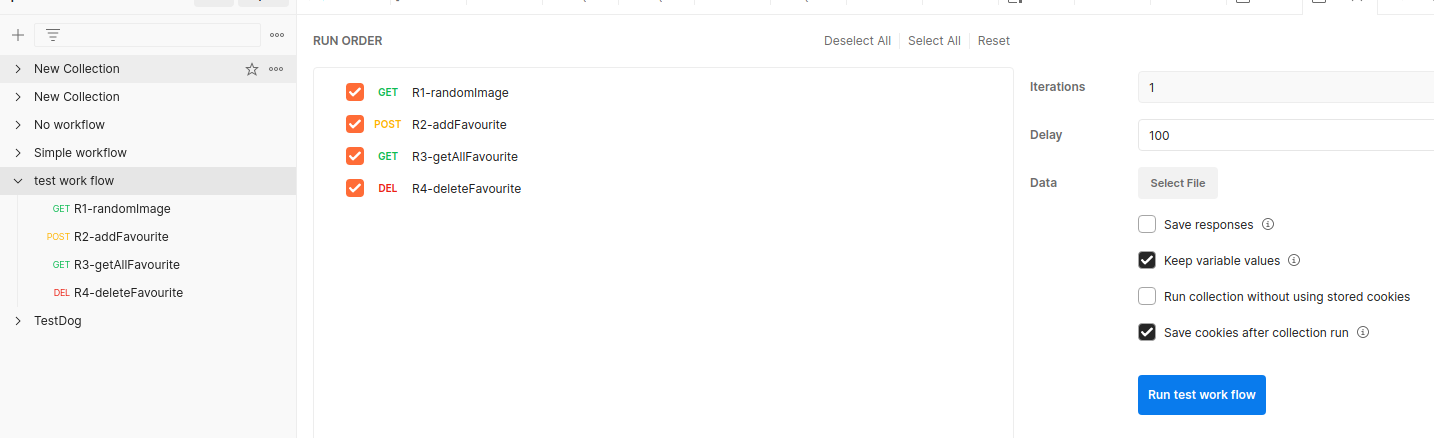


**2.5 Thực hiện chạy collection để kiểm chứng workflow thông qua Collection Runner.**

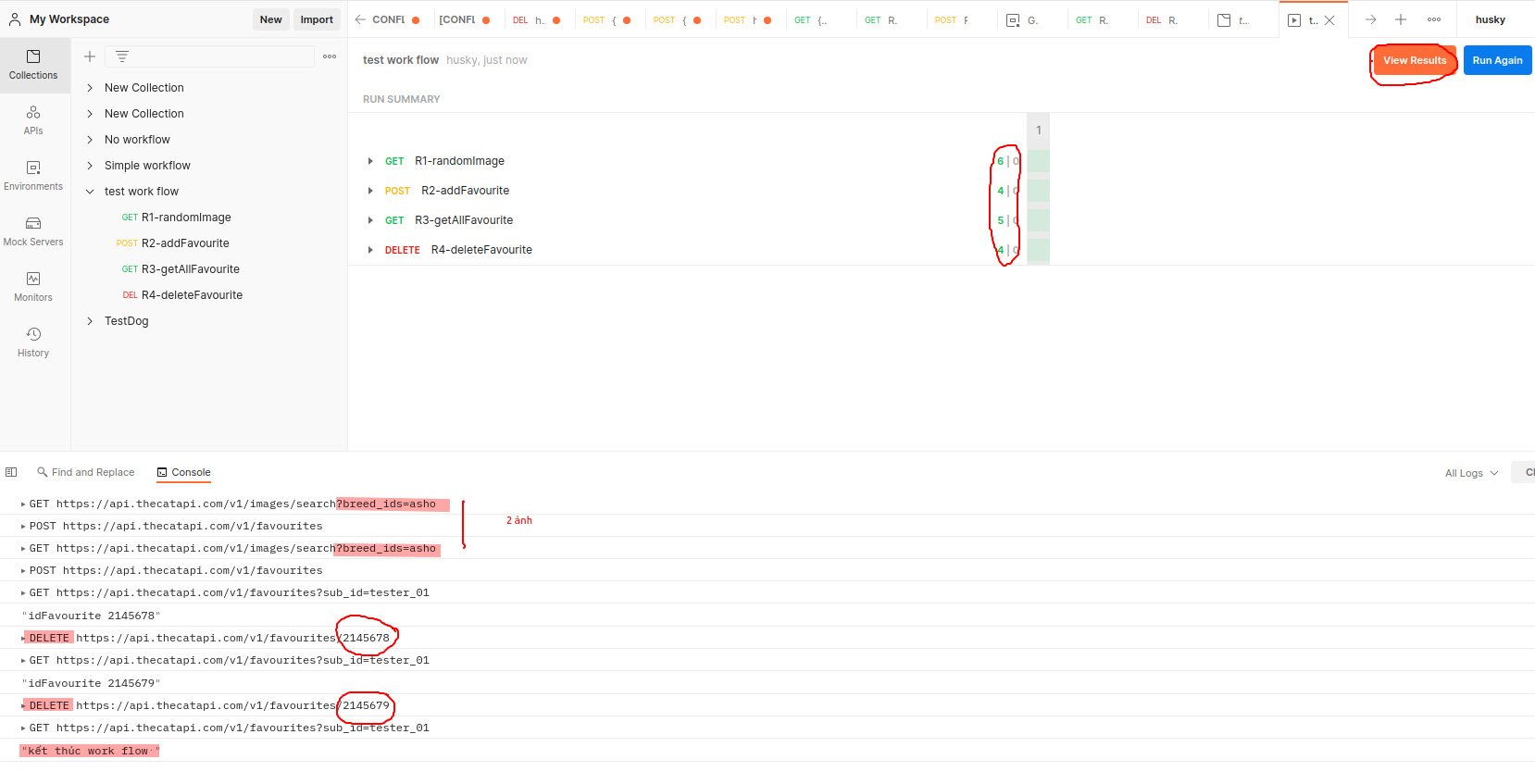
Xóa các biến môi trường trước khi chạy collection.



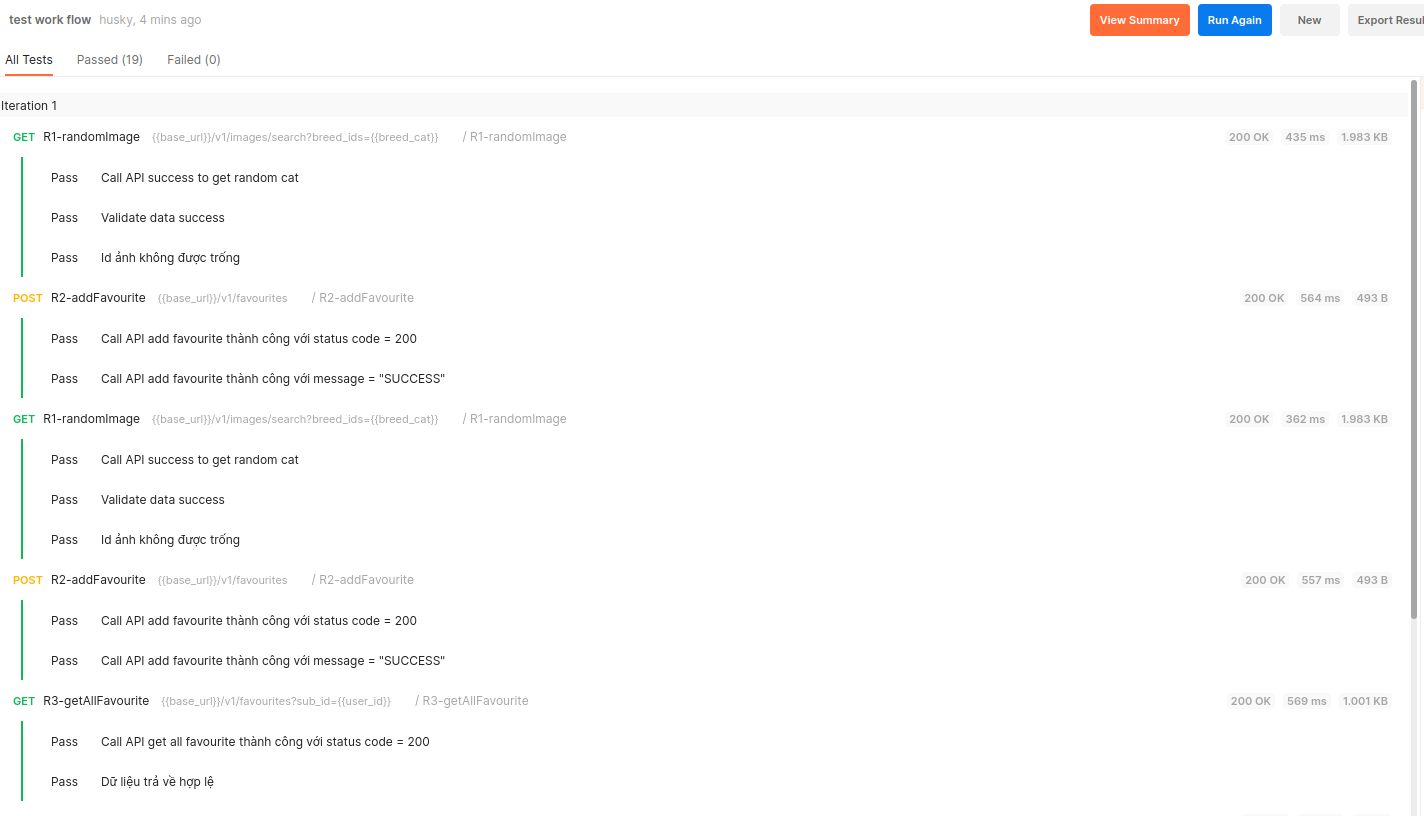
Thực thi Collection.

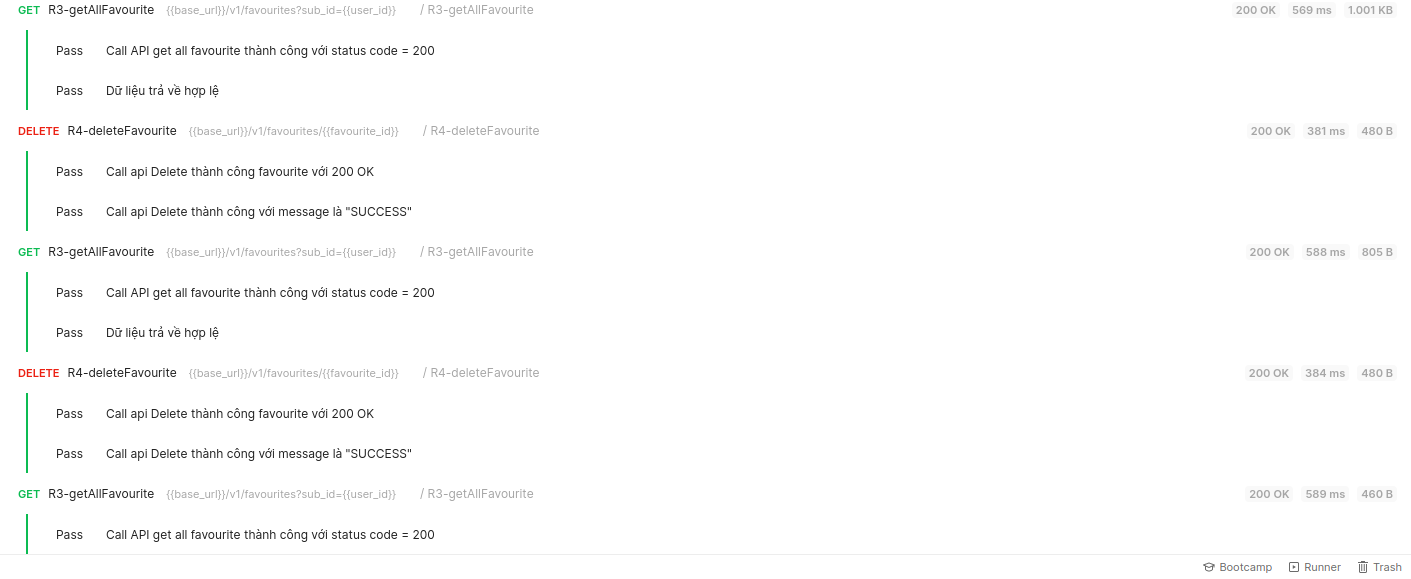


Các bạn có thể xem luồng gọi API ở cửa sổ Console.



View Sumary.





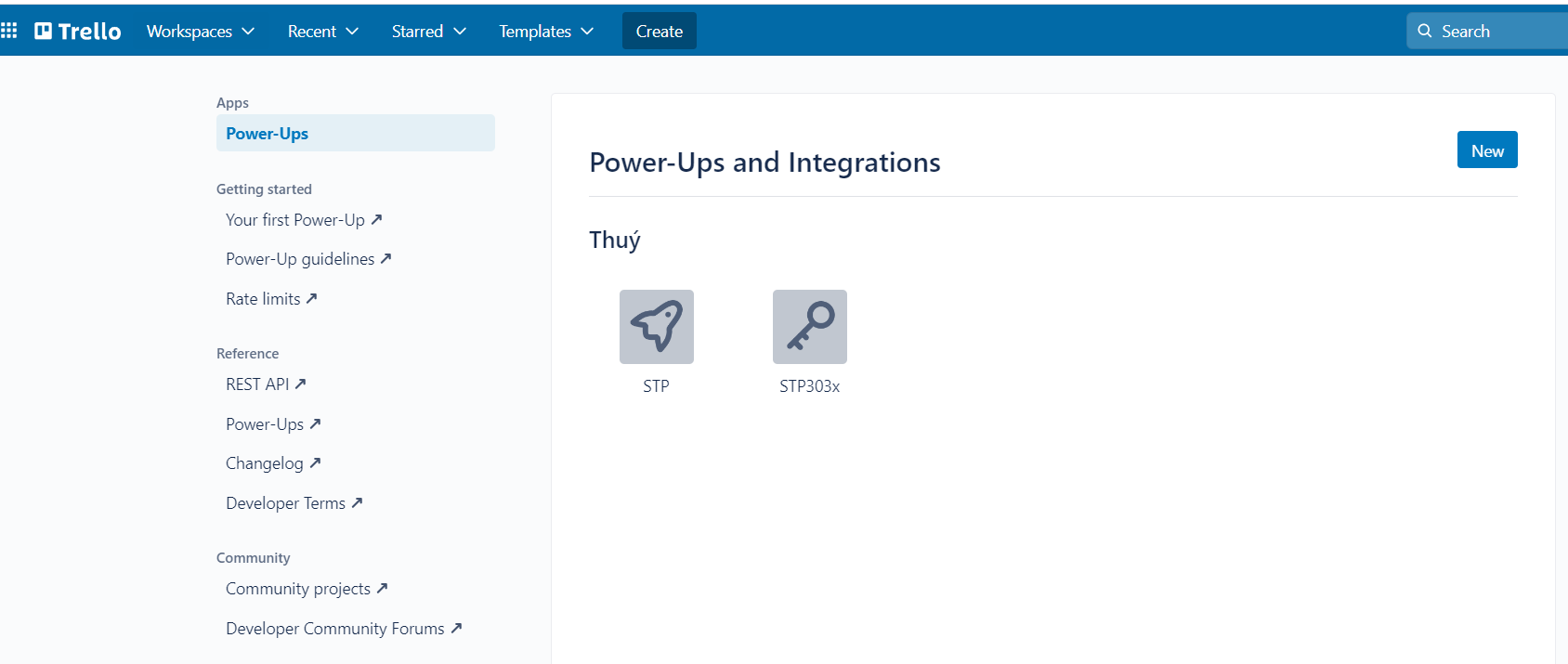
**III/ Học viên thực hành.**

Phần này, chúng ta sẽ quay lại luyện tập với Trello (Các bạn có thể tham khảo nội dung bài lab số 8 để hiểu hơn về Trello)

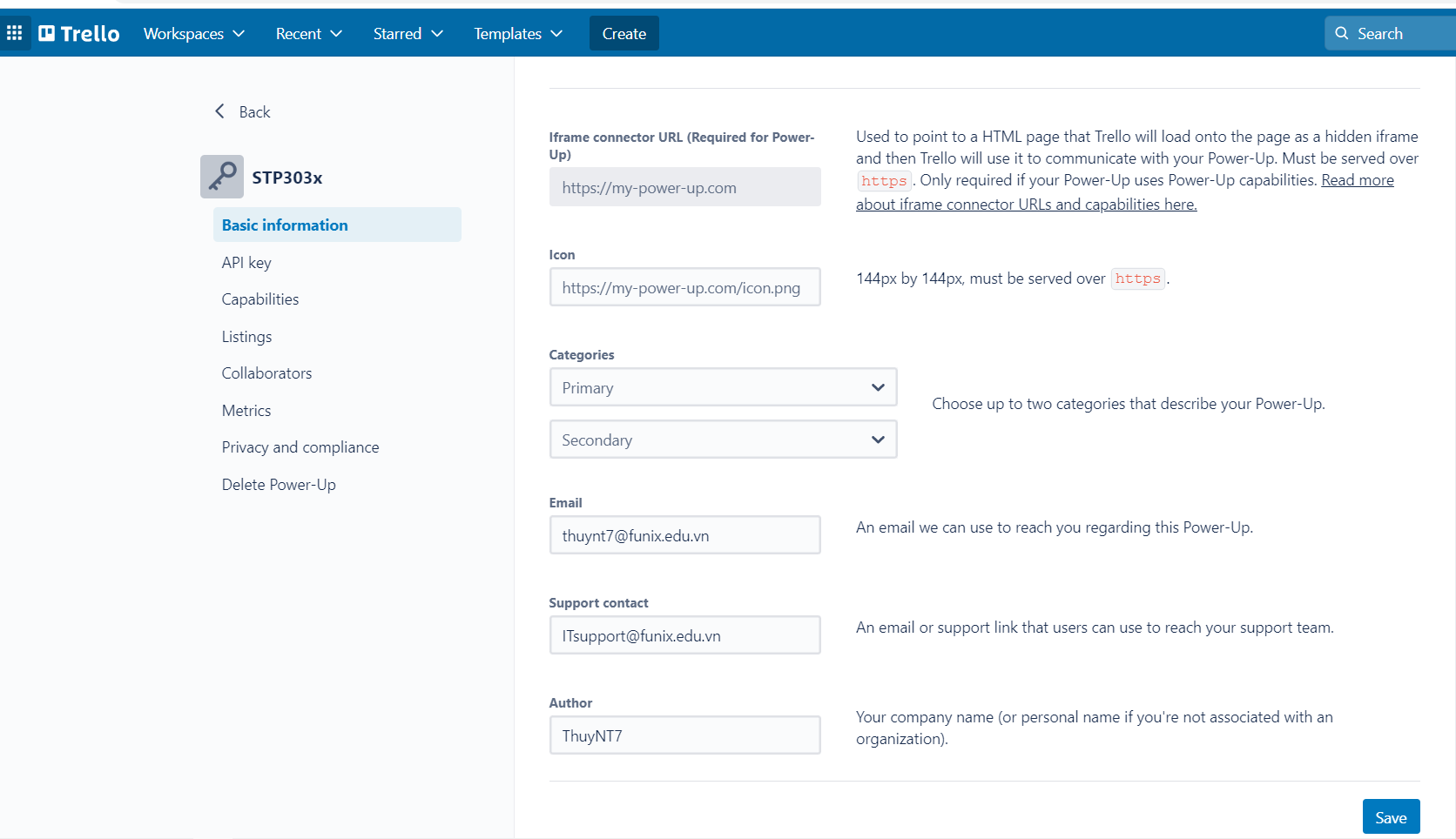
**Điều kiện thực hành.**

Các API của Trello đều yêu cầu xác thực qua **key** và **token**, do đó bạn cần đăng kí tài khoản trên trang chủ của Trello: <https://trello.com/login>

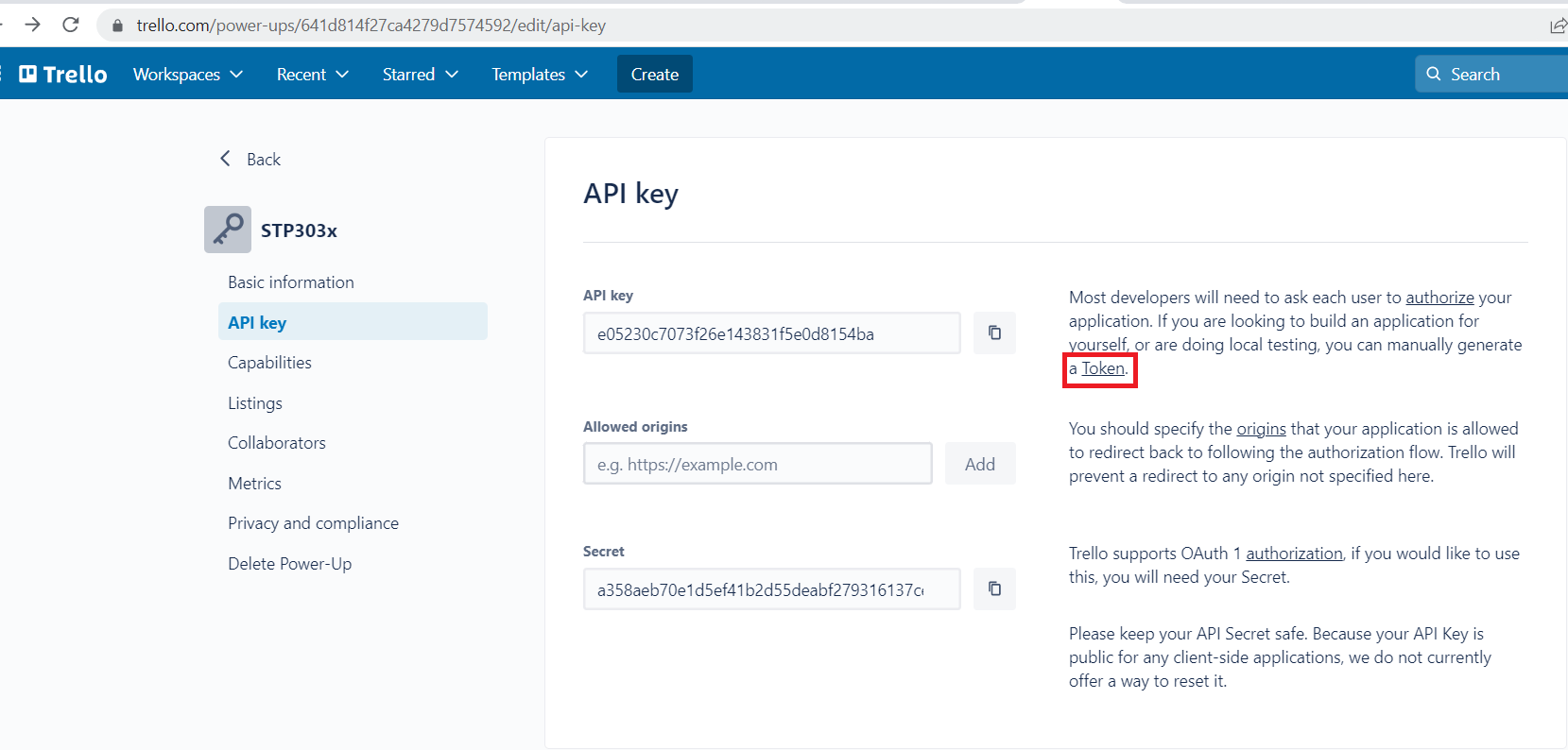
* Truy cập: <https://trello.com/login> để tạo tài khoản Trello.  
  Sau khi tạo xong tài khoản, chúng ta cần thực hiện đăng nhập.
* Truy cập : <https://trello.com/power-ups/admin/> để lấy **key và token.**
* Nhấn vào tạo mới (NEW) ở chỗ Power-Ups and Integrations như hình dưới đây:



* Sau đó, điền đầy đủ thông tin trong đó => Save



* Sau đó nó sẽ chuyển người dùng sang trang mới Power-ups administrator ở đó nó sẽ cung cấp api key => sau đó nhấn vào Token

****

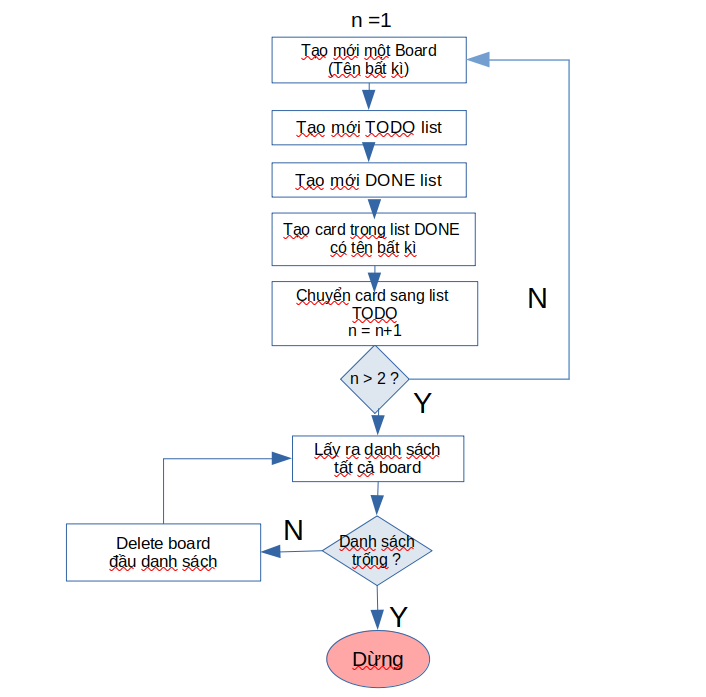
|  |  |
| --- | --- |

Các bạn cần lưu lại **key** và **token** bảo mật để sử dụng kiểm thử với API của trello.

**Danh sách API.**

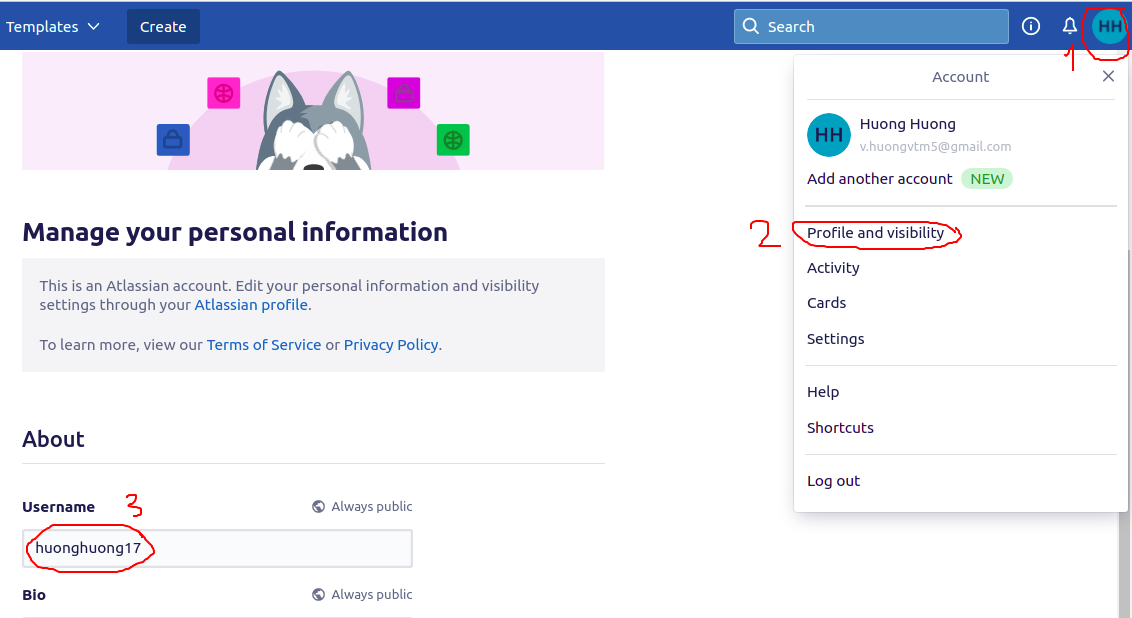
|  | **API** | **Tham số** | **Yêu cầu** |
| --- | --- | --- | --- |
| 1 | Tạo một Board.  **POST**: *https://api.trello.com/1/boards* | Body:  "**name**":required  "defaultLists":optional  {Boolean: true|false}  Default true, mặc định Trello sẽ tạo ra 3 list cho 1 board (To Do, Doing, Done) | - API yêu cầu xác thực bằng key và token Nếu nhập sai hoặc để trống key hoặc token: API trả về **401 Unauthorized.**  - Nếu không nhập name thì API trả về **400 Bad request.** -Tạo Board thành công, API trả về **200 OK** |
| 2 | Tạo một list **POST** *https://api.trello.com/1/lists* | Body: "name":required  "idBoard": required  (Id của board chứa list - lấy từ response của API tạo board) | - API yêu cầu xác thực bằng key và token Nếu nhập sai hoặc để trống key hoặc token: API trả về **401 Unauthorized.**  - Nếu không nhập name hoặc idBoard thì API trả về **400 Bad request.**  - Nếu không đúng idBoard thì API trả về **400 Bad request**  -Tạo List thành công, API trả về **200 OK** |
| 3 | Tạo một Card **POST** *https://api.trello.com/1/cards* | "name":required  "idList": required  (Id của List - lấy từ response của API tạo List) "desc": optional (mô tả chi tiết thông tin card) | - API yêu cầu xác thực bằng key và token Nếu nhập sai hoặc để trống key hoặc token: API trả về **401 Unauthorized.**  - Nếu không nhập name hoặc idList thì API trả về **400 Bad request.**  - Nếu không nhập hoặc nhập sai **idList** thì API trả về **400 Bad request**  -Tạo Card thành công, API trả về **200 OK** |
| 4 | Update Card  **PUT**  *https://api.trello.com/1/cards/{cardId}* | **Path param:** "cardId": id của card (required)  **Query param;** "name": tên card  "closed": trạng thái card (true|false)  "idList": id của list chứa card.  "idBoard": id của board chứa card | Update thành công API trả về **200 0K**.  Update thất bại:  Nếu không tìm thấy **cardId** APi cần trả về **404 not found**.  Nếu không tìm **name** trống, API cần trả về **400 bad request.**  Nếu idList hoặc idBoard không tìm thấy, API cần trả về **400 bad request.** |
| 5 | Lấy ra danh sách tất cả các board thuộc về người dùng.  **GET**  *https://api.trello.com/1/members/{id}/boards* | **Path param:** "id": id ở đây là username của tài khoản.  **Response: mảng các board**  [{ "id": id board "name": tên board  "closed": trạng thái board (true|false)  "url": Link đến board card.  ….  }, …. ] | Call API thành công, api trả về **200 OK**  Thông tin trả về là 1 danh sách các board.  Nếu id user nhập vào không đúng, hoặc không tồn tại, API cần trả về **404 Not Found.** |
| 6 | Delete một Board.  **DELETE** *https://api.trello.com/1/boards/{id}* | **Path param:** "id": id ở đây là id của board | Delete thành công , API trả về **200 OK**  Nếu id board không đúng thì trả về **404 not found**.  Các trường hợp code khác **200 OK => thất bại** |

**WorkFlow.**



**Chú ý:**   
Tại API số 5, **id** của user chính là **username** của các bạn.  
  
Các bạn có thể đăng nhập vào : <https://trello.com/> chọn tài khoản góc phải màn hình.  
Sau đó lưu lại thông tin username của các bạn để phục vụ test APi số 5.

**Minh họa**

**Gợi ý làm bài:**

1. Tạo các biến global, environment để lưu thông tin như : url, key, token , username…

2. Viết Script kiểm thử API theo yêu cầu.

3. Thiết lập động các biến để lưu thông tin id cần thiết như id boad, list, card

4. Chú ý áp dụng API số 4 để có thể chuyển một card sang list khác bằng cách thay đổi tham số **idList** thành id của List muốn chuyển sang.

5. Lưu các request vào một collection

6. Trong mỗi request, sử dụng các điều kiện để đáp ứng workflow.