HỌC VIỆN KỸ THUẬT MẬT MÃ

KHOA CÔNG NGHỆ THÔNG TIN

***

BÁO CÁO KIỂM THỬ PHẦN MỀM NHÚNG

Đề Tài: Kiểm thử API bằng công cụ Postman trên Thingsboard

Giáo viên hướng dẫn: ThS. Thái Thị Thanh Vân

Sinh viên thực hiện: Nhóm 9

1. Hoàng Thị Hường CT040426
2. Nguyễn Thị Nhị CT040435
3. Tạ Thị Thơm CT040447

Hà Nội 2023

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

LỜI MỞ ĐẦU

Trong thời đại công nghệ thông tin hiện nay, công nghệ phần mềm đang phát triển với tốc độ vượt bậc, đặc biệt là trong lĩnh vực phát triển phần mềm. Việc xây dựng phần mềm đã trở nên đơn giản hơn nhiều nhờ vào sự hỗ trợ của nhiều công cụ tiên tiến. Tuy nhiên, do độ phức tạp của phần mềm và những giới hạn về thời gian và chi phí, việc đảm bảo chất lượng phần mềm là một thách thức lớn đối với các nhà phát triển.

Kiểm thử phần mềm là một quá trình quan trọng trong việc đảm bảo chất lượng phần mềm. Nó là một quá trình liên tục, được thực hiện trong mọi giai đoạn phát triển phần mềm. Tuy nhiên, việc kiểm thử phần mềm tốn kém, mất thời gian và khó phát hiện được hết lỗi. Do đó, để đảm bảo chất lượng phần mềm, cần có một chiến lược phù hợp và quản lí chặt chẽ. Với việc ứng dụng IoT đang ngày càng phổ biến, việc kiểm thử phần mềm chạy trên nền tảng này là rất quan trọng. Vì vậy, việc nghiên cứu về cách kiểm thử phần mềm trên nền tảng IoT là rất cần thiết.

Để có thể giải quyết được vấn đề đã nêu ở trên, nhóm đã thảo luận và đưa ra đề tài "***Kiểm thử API bằng công cụ Postman trên Thingsboard***". Mục tiêu của đề tài là tìm hiểu về kiểm thử API sử dụng công cụ Postman trên website nói chung và trên Thingsboard nói riêng. Nghiên cứu này cũng nhằm triển khai công cụ kiểm thử phần mềm tự động để giảm nhân lực kiểm thử và đảm bảo chất lượng phần mềm hơn với công việc kiểm thử thủ công. Với mục tiêu đặt ra như vậy, những nội dung và kết quả nghiên cứu chính của đề tài được trình bày trong ba chương như sau:

Chương 1: KIỂM THỬ API BẰNG CÔNG CỤ POSTMAN

Chương 2: XÂY DỰNG KẾ HOẠCH KIỂM THỬ

Chương 3: TIẾN HÀNH CÀI ĐẶT VÀ DEMO THỰC NGHIỆM VỚI CÔNG CỤ POSTMAN

Trong quá trình thực hiện đề tài, do thời gian cũng như trình độ của chúng em còn có những hạn chế nhất định nên không thể tránh khỏi những sai sót. Rất mong được sự góp ý của các thầy cô, cũng như các bạn học viên để bài báo cáo này được hoàn thiện hơn.

CHƯƠNG 1: KIỂM THỬ API BẰNG CÔNG CỤ POSTMAN

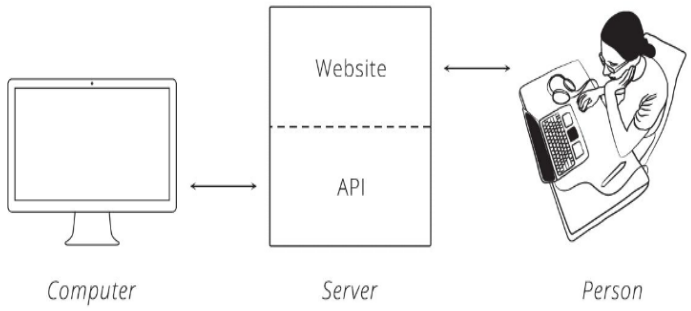
1.1 API

1.1.1 API là gì?

API là chữ viết tắt của "Application Programming Interface" (giao diện lập trình ứng dụng), là một phương tiện để các phần mềm, ứng dụng hoặc hệ thống khác có thể tương tác và trao đổi dữ liệu với nhau. API là một tập hợp các quy tắc, giao thức và công nghệ được thiết kế để cho phép hai ứng dụng khác nhau liên lạc và chia sẻ thông tin với nhau.

Các API cho phép các nhà phát triển tạo ra các ứng dụng có thể kết nối và sử dụng các dịch vụ, chức năng hoặc dữ liệu từ các hệ thống khác nhau. Nó cũng cho phép các nhà phát triển sử dụng các thư viện và framework có sẵn để phát triển ứng dụng nhanh hơn.

Nói đơn giản, API là cái cầu nối giữa client và server. Client ở đây có thể là máy tính, điện thoại sử dụng hệ điều hành khác nhau và được viết bằng những ngôn ngữ khác nhau, ví dụ như Swift, Objective-C, Java. Tương tự, server back-end cũng được viết bằng các ngôn ngữ khác nhau. Để hai đối tượng này có thể nói chuyện được với nhau chúng phải nói cùng một ngôn ngữ. Ngôn ngữ ấy chính là API.



Hình 1. 1: Hoạt động của API

Vì sao phải kiểm thử API?

Kiểm thử API là rất quan trọng vì nhiều lý do:

- *Đảm bảo chức năng*: API được thiết kế để thực hiện các tác vụ hoặc chức năng cụ thể, vì vậy kiểm thử API đảm bảo rằng chúng hoạt động đúng như ý định.

Nếu có bất kỳ vấn đề gì, nó có thể được phát hiện sớm trong quá trình phát triển và sửa chữa trước khi ảnh hưởng đến người dùng cuối.

- *Phát hiện lỗi*: API thường được sử dụng bởi các ứng dụng và nền tảng khác nhau. Kiểm thử API có thể giúp phát hiện bất kỳ lỗi hoặc vấn đề nào có thể phát sinh khi tích hợp với các hệ thống khác. Tìm thấy những lỗi này sớm có thể giúp ngăn chặn chúng gây ra những vấn đề lớn hơn sau này.
- *Bảo mật*: API có thể là mối đe dọa bảo mật nếu chúng không được kiểm thử đúng cách. Tin tặc có thể khai thác các lỗ hổng trong API để truy cập thông tin nhạy cảm hoặc thực hiện các hoạt động độc hại. Bằng cách kiểm thử API, các nhà phát triển có thể xác định và khắc phục các lỗ hổng bảo mật trước khi chúng có thể bị khai thác.
- *Hiệu suất*: API có thể ảnh hưởng đến hiệu suất của một ứng dụng. Kiểm thử API có thể giúp xác định bất kỳ vấn đề hiệu suất nào có thể phát sinh, chẳng hạn như thời gian phản hồi chậm hoặc sử dụng mạng cao.
- *Tài liệu*: Kiểm thử API có thể giúp đảm bảo tài liệu chính xác và cập nhật. Điều này quan trọng đối với các nhà phát triển tích hợp API, vì tài liệu chính xác có thể giúp họ sử dụng API hiệu quả hơn.

1.1.2 Giao thức sử dụng trong API

Giao thức chính là những luật lệ được chấp thuận để hai máy tính có thể nói chuyện với nhau. Để hai máy tính giao tiếp hiệu quả, server phải biết chính xác cách mà client sắp xếp cái message nó gửi lên như thế nào. Chúng ta đã từng nghe đến những Protocol cho những mục đích khác nhau, ví dụ như Mail có POP hay IMAP, message có XMPP, kết nối thiết bị: Bluetooth. Trong web thì protocol chính là HTTP – HyperText Transfer Protocol, vì sự phổ biến của nó mà hầu hết các công ty chọn nó là giao thức cho các API.

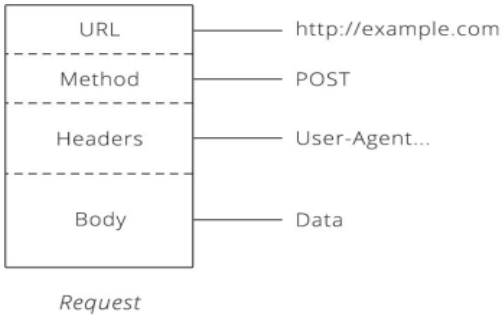
❖ *Hoạt động của giao thức http:*

Bao gồm hai thành phần chính: request và response.

Http hoạt động dựa trên **mô hình Client (máy khách) – Server (máy chủ)**.

Các máy tính của người dùng sẽ đóng vai trò làm máy khách (Client).

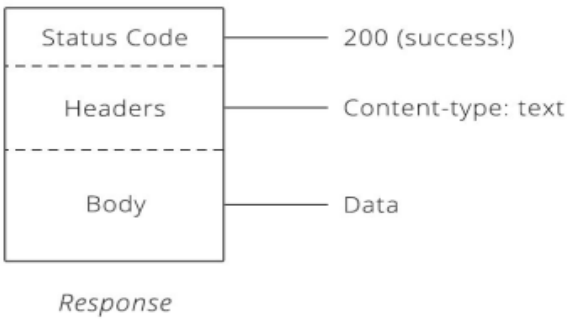
➢ Request:



Hình 1. 2: Request trong API

- *URL* là một địa chỉ duy nhất (dùng danh từ), có thể là web page, image, hoặc video. API mở rộng cái ý tưởng gốc của URL cho những thứ khác, ví dụ: customers, products. Và như thế client dễ dàng cho server biết cái nó muốn là cái gì, những cái này còn được gọi chung là "resources" – nguồn lực.
- *Method*: là cái hành động client muốn tác động lên "resources", và nó thường là động từ. Có 4 loại Method hay được dùng: GET, POST, PUT, DELETE.
- *Header*: nơi chứa các thông tin cần thiết của một request nhưng end-users không biết có sự tồn tại của nó. Ví dụ: độ dài của request body, thời gian gửi request, loại thiết bị đang sử dụng, loại định dạng cái response mà client đọc được…
- *Body*: nơi chứa thông tin mà client sẽ điền.

➢ Response:



Hình 1. 3: Response trong API

- *Status code* là những con số có 3 chữ số và có duy nhất một ý nghĩa. Ví dụ Error "404 Not Found" hoặc "503 Service Unavailable".
- *Header và Body* tương đối giống với request.

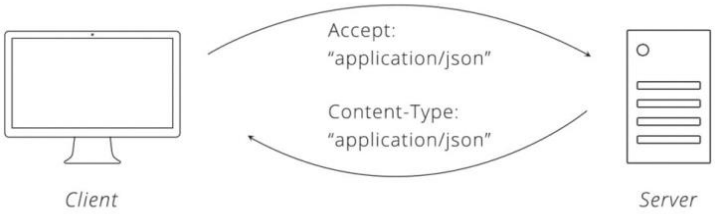
1.1.3 Định dạng dữ liệu

JSON (JavaScript Object Notation) được sử dụng nhiều trong Restful API. Nó được xây dựng từ Javascript, ngôn ngữ mà được dùng nhiều, tương thích với cả front-end và back-end của cả web app và web service. JSON là một định dạng đơn giản với 2 thành phần: keys và values.

- Key thể hiện thuộc tính của Object
- Value thể hiện giá trị của từng Key

  Ví dụ:

Dựa vào 2 thành phần Content-Type và Accept, client và server có thể hiểu và làm việc một cách chính xác.



Hình 1. 4: Content-Type

1.2 Giới thiệu về POSTMAN

Postman là một ứng dụng dùng để phát triển và kiểm thử các API. Với Postman, chúng ta có thể tạo ra các request (yêu cầu) tới các API, sau đó chạy các test (kiểm tra) để đảm bảo rằng API đang hoạt động đúng cách. Đây là một công cụ rất hữu ích cho các nhà phát triển, nhà kiểm thử và những người quản lý API.

Postman hỗ trợ nhiều loại request như GET, POST, PUT, DELETE và nhiều loại request khác. Chúng ta có thể thiết lập các thông số cho các request này, bao gồm *headers, parameters, body, authentication* và các thuộc tính khác. Ngoài ra, Postman còn cung cấp tính năng **automation** để giúp tự động hóa các bước kiểm thử. Chúng ta có thể tạo ra các script (kịch bản) để kiểm tra các phản hồi từ API và đưa ra quyết định dựa trên kết quả đó.

Postman cũng cho phép chúng ta chia sẻ các collection (tập hợp) của request và test với các đồng nghiệp của mình. Điều này giúp cho các nhóm phát triển và kiểm thử có thể làm việc chung và chia sẻ thông tin một cách hiệu quả.

Cuối cùng, Postman còn có tính năng monitoring, cho phép chúng ta theo dõi hiệu suất của các API theo thời gian thực. Tính năng này cho phép theo dõi các metric (đo lường) như thời gian phản hồi, thời gian xử lý, lỗi và các thuộc tính khác của API.

1.3 Các tính năng chính của POSTMAN

- *Tạo và gửi các yêu cầu HTTP*: POSTMAN cho phép người dùng tạo và gửi các yêu cầu HTTP, bao gồm các phương thức GET, POST, PUT, DELETE, PATCH và nhiều hơn nữa. Người dùng có thể thêm các thông số và tham số cho mỗi yêu cầu và gửi chúng đến máy chủ web.
- *Kiểm tra phản hồi từ máy chủ*: POSTMAN cung cấp các công cụ để người dùng kiểm tra phản hồi từ máy chủ web, bao gồm mã trạng thái HTTP, thông tin phản hồi và các thông số khác nhau. Người dùng có thể xem dữ liệu phản hồi dưới dạng văn bản hoặc JSON.
- *Quản lý các yêu cầu HTTP*: POSTMAN cho phép người dùng lưu trữ các yêu cầu HTTP của mình, tổ chức chúng vào các bộ sưu tập và quản lý chúng dễ dàng. Người dùng có thể sắp xếp các yêu cầu theo thư mục, thêm chú thích và thêm thông tin chi tiết cho từng yêu cầu.
- *Kiểm tra API*: POSTMAN cung cấp các công cụ để kiểm tra API của người dùng bằng cách tạo các kịch bản thử nghiệm và bộ kiểm tra tự động. Người

dùng có thể kiểm tra các định dạng dữ liệu khác nhau, tạo các kịch bản thử nghiệm phức tạp và đo lường hiệu suất API.

- *Xử lý các biến môi trường*: POSTMAN cho phép người dùng xử lý các biến môi trường, cho phép quản lý các giá trị cấu hình khác nhau và dễ dàng chuyển đổi giữa chúng. Điều này giúp người dùng quản lý các yêu cầu API trong nhiều môi trường khác nhau, chẳng hạn như môi trường phát triển và môi trường sản xuất.
- *Lưu trữ thông tin xác thực*: POSTMAN cho phép người dùng lưu trữ các thông tin xác thực như mã thông báo truy cập hoặc thông tin đăng nhập để dễ dàng sử dụng chúng cho các yêu cầu khác.
- *Chia sẻ và hợp tác:* POSTMAN cho phép người dùng chia sẻ bộ sưu tập yêu cầu của họ với những người khác và làm việc chung trên cùng một tài liệu. Điều này giúp tăng tính hiệu quả của việc phát triển ứng dụng và đồng thời giúp tiết kiệm thời gian cho các thành viên trong nhóm.
- *Mở rộng bằng các phần mở*: POSTMAN có thể được mở rộng bằng các phần mở để bổ sung các tính năng mới hoặc tùy chỉnh theo nhu cầu của người dùng. Các phần mở này có thể được tìm thấy trên trang web chính thức của POSTMAN hoặc từ cộng đồng người dùng.
- *Tích hợp với các công cụ khác*: POSTMAN có thể tích hợp với các công cụ khác, bao gồm các công cụ quản lý phiên bản, công cụ quản lý môi trường, các công cụ theo dõi lỗi và các công cụ kiểm tra hiệu suất để tăng tính linh hoạt và hiệu quả cho các dự án phát triển.

1.4 Làm việc với Postman

Để làm việc với Postman, cần làm theo các bước sau:

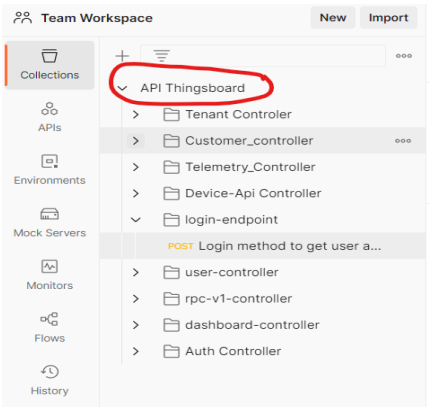
- *Cài đặt POSTMAN:* Đầu tiên, cần tải và cài đặt Postman từ trang chủ của nó. Sau khi cài đặt, khởi chạy Postman và bắt đầu làm việc.
- *Tạo yêu cầu API*: Để tạo yêu cầu API, cần chọn phương thức yêu cầu (GET, POST, PUT, DELETE), nhập URL của API và thêm các tham số, tiêu đề và thân yêu cầu nếu cần thiết.

- *Chạy yêu cầu API:* Sau khi tạo yêu cầu API, có thể chạy nó để xem kết quả trả về của API. Postman sẽ hiển thị phản hồi của API trong tab Response. Tiến hành kiểm tra xem phản hồi đó có đúng hay không và sửa chữa nếu cần thiết.
- *Quản lý biến và môi trường*: Postman cho phép quản lý các biến và môi trường để tùy chỉnh yêu cầu API và dễ dàng sử dụng lại trong các bộ kiểm thử khác nhau. Có thể định nghĩa các biến trong một môi trường và sử dụng chúng trong các yêu cầu API bằng cách sử dụng cú pháp {{variable_name}}.
- *Kiểm tra độ bảo mật của API*: Có thể kiểm tra độ bảo mật của API bằng cách định nghĩa các chế độ xác thực và tạo yêu cầu API với các thông tin xác thực đó.
- *Kiểm tra hiệu suất của API:* Postman cho phép kiểm tra hiệu suất của API bằng cách gửi yêu cầu API liên tiếp và xem thời gian phản hồi của API.
- *Lưu trữ yêu cầu API:* Postman cho phép lưu trữ các yêu cầu API để sử dụng lại trong tương lai hoặc chia sẻ với các thành viên trong nhóm.
- *Sử dụng bộ kiểm thử:* Postman cũng cung cấp bộ kiểm thử để tự động kiểm thử API với các bộ dữ liệu khác nhau.

1.5 Quy trình làm việc của Postman

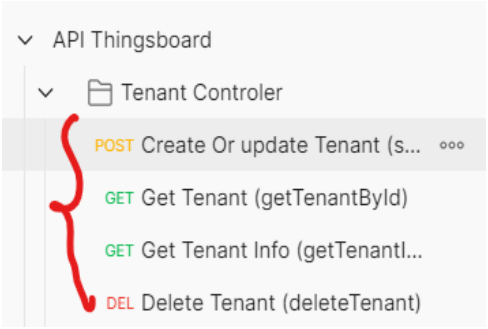
Postman là một công cụ quản lý và kiểm thử API. Quy trình làm việc của Postman thường bao gồm các bước sau:

- *Tạo một bộ sưu tập* (collection): Bộ sưu tập chứa tất cả các yêu cầu API cần thiết để thực hiện một tác vụ cụ thể.



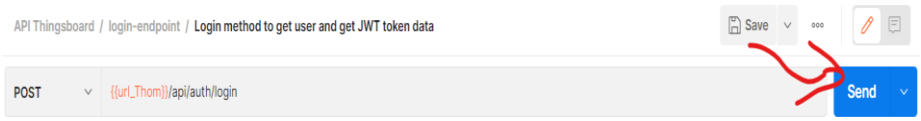
Hình 1. 5: Tạo bộ sưu tập trong Postman

- *Thêm các yêu cầu API* (requests) vào bộ sưu tập: Chúng ta có thể thêm các yêu cầu API vào bộ sưu tập bằng cách nhập URL và cấu hình các thông số yêu cầu như phương thức, tham số và tiêu đề.



Hình 1. 6: Tạo request trong collection

- *Chạy các yêu cầu API*: Sau khi thêm yêu cầu API vào bộ sưu tập, có thể chạy chúng để kiểm tra tính đúng đắn và đáp ứng của API.

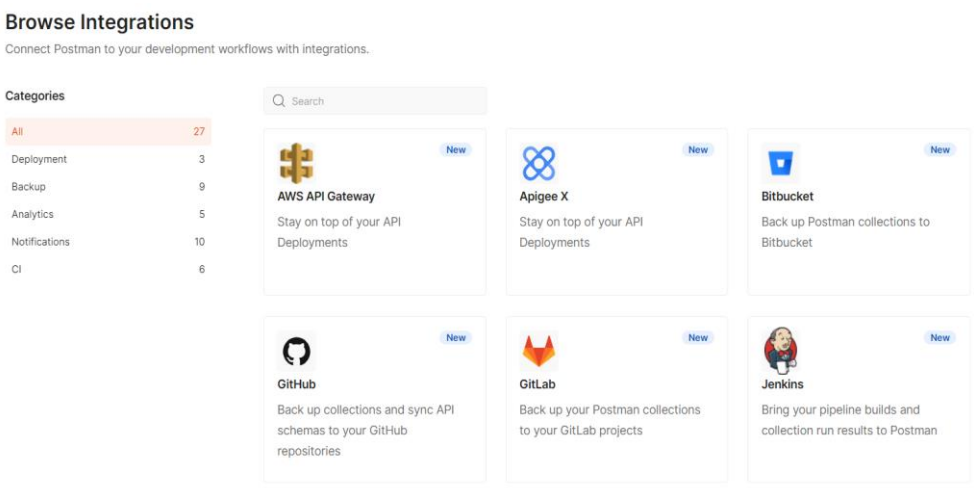


Hình 1. 7: Run API

- *Kiểm tra kết quả:* Sau khi chạy yêu cầu API, Postman sẽ trả về các kết quả trong cửa sổ xem trước (preview). Chúng ta có thể kiểm tra các thông tin trả về bao gồm mã trạng thái, phản hồi và thời gian phản hồi. Xem phản hồi ở chế độ "pretty" sẽ dễ theo dõi hơn.

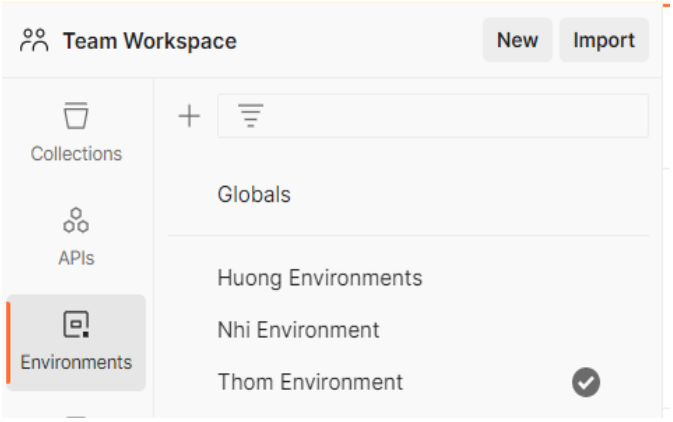
Hình 1. 8: Phản hồi trả về của API

- *Tích hợp với các công cụ khác:* Postman cho phép tích hợp với các công cụ khác như Git, Jenkins, Newman, để thực hiện các tác vụ như chia sẻ bộ sưu tập và chạy tự động các yêu cầu API.



Hình 1. 9: Các công cụ tích hợp trong Postman

- *Tạo môi trường (environment)*: Postman cung cấp một chức năng để tạo môi trường, cho phép cấu hình các biến môi trường như URL và tham số. Chúng ta có thể tạo nhiều môi trường để kiểm tra API trên các môi trường khác nhau.



Hình 1. 10: Tạo môi trường kiểm thử

1.6 So sánh các công cụ kiểm thử tự động

Bảng 1. 1: Bảng so sánh các công cụ kiểm thử tự động

Tên công cụ	Ưu điểm	Nhược điểm
Selenium	Selenium là một công cụ phổ biến và mạnh mẽ để kiểm thử ứng dụng web. Nó cho phép kiểm thử trực tiếp trên trình duyệt và hỗ trợ nhiều ngôn ngữ lập trình.	Selenium không hỗ trợ kiểm thử các API REST, chỉ hỗ trợ kiểm thử ứng dụng web.

JMeter	JMeter là một công cụ kiểm thử hiệu năng mạnh mẽ và phổ biến, hỗ trợ nhiều giao thức và cung cấp một giao diện đơn giản để cấu hình các kịch bản kiểm thử.	JMeter có thể khó sử dụng đối với người mới bắt đầu và cần một số kiến thức về lập trình.
Katalon Studio	Katalon Studio là một công cụ kiểm thử tự động đa năng, có thể kiểm thử các ứng dụng web, desktop và mobile.	Katalon Studio có giới hạn trong phiên bản miễn phí và cần trả phí để sử dụng các tính năng đầy đủ.
Postman	Postman là một công cụ kiểm thử tự động, có giao diện thân thiện, dễ sử dụng, hỗ trợ kiểm thử API đa dạng, tích hợp nhiều tính năng hữu ích và tài liệu hướng dẫn đầy đủ.	Postman có yêu cầu tài nguyên máy tính khá cao, không hỗ trợ kiểm thử đơn vị và phải trả phí để sử dụng các tính năng cao cấp.

Bảng trên đưa ra những ưu và nhược điểm của một số công cụ kiểm thử tự động API phổ biến. Trong quá trình tìm hiểu, nhóm đã quyết định lựa chọn công cụ kiểm thử tự động API là: Postman. Postman là một công cụ kiểm thử API đáng tin cậy với nhiều ưu điểm như:

- *Dễ sử dụng*: Postman có giao diện đơn giản, thân thiện và dễ sử dụng. Người dùng không cần có kiến thức sâu về lập trình để bắt đầu sử dụng nó.
- *Đa nền tảng:* Postman có thể chạy trên nhiều nền tảng khác nhau, bao gồm Windows, Mac OS và Linux.
- *Kiểm thử API đầy đủ:* Postman cung cấp nhiều tính năng để kiểm tra các API như xác thực, quản lý biến môi trường, kiểm thử chức năng, kiểm thử tính toàn vẹn, kiểm thử hiệu năng và nhiều hơn nữa.

- *Tích hợp với nhiều công cụ:* Postman tích hợp với nhiều công cụ phát triển phổ biến khác như Jenkins, Swagger và Git.
- *Hỗ trợ đầy đủ cho REST API*: Postman có thể kiểm thử REST API một cách dễ dàng và hiệu quả.
- *Đội ngũ hỗ trợ và cộng đồng lớn:* Postman có một đội ngũ hỗ trợ chuyên nghiệp và cộng đồng lớn của các nhà phát triển sử dụng công cụ này.

Tuy có nhiều ưu điểm nhưng công cụ này cũng có một số nhược điểm như: bị giới hạn trong phiên bản miễn phí, khó khăn trong việc quản lý các kịch bản kiểm thử lớn, … Nhưng trong phạm vi và cũng như mục đích sử dụng cho bài báo cáo này, nhóm nhận thấy rằng công cụ POSTMAN là phù hợp nhất.

CHƯƠNG 2: XÂY DỰNG KẾ HOẠCH KIỂM THỬ

2.1 Giới thiệu

2.1.1 Mục đích

Kiểm thử API trên Thingsboard đảm bảo rằng API được phát triển đáp ứng được các yêu cầu và đưa ra các kết quả đúng đắn. Đảm bảo rằng API hoạt động một cách chính xác và liên tục, giảm thiểu sự cố và tăng tính ổn định của hệ thống. Tăng khả năng phát hiện và sửa lỗi trước khi triển khai API cho người dùng sử dụng.

2.1.2 Tổng quan về Thingsboard

Thingsboard là một nền tảng quản lý dữ liệu IoT (Internet of Things) mã nguồn mở được thiết kế để thu thập, xử lý và hiển thị dữ liệu từ các thiết bị IoT khác nhau. Với Thingsboard, người dùng có thể giám sát và điều khiển các thiết bị IoT từ xa, lưu trữ và phân tích dữ liệu và đưa ra các hành động phù hợp.

Thingsboard được xây dựng trên nền tảng Java và sử dụng các công nghệ khác như AngularJS, Cassandra và Kafka để cung cấp các tính năng đa dạng. Nền tảng này hỗ trợ nhiều giao thức IoT phổ biến như MQTT, CoAP và HTTP, giúp kết nối đến các thiết bị IoT từ nhiều nhà sản xuất khác nhau. Ngoài ra, Thingsboard còn tích hợp các tính năng bảo mật mạnh mẽ như xác thực người dùng, phân quyền, mã hóa dữ liệu và giám sát tấn công.

Các tính năng của Thingsboard

- *Bảo mật*: Hỗ trợ cung cấp và quản lý các thiết bị có quản lý thông tin đăng nhập. Các quy tắc bảo mật tùy chỉnh cho từng giao thức có thể được áp dụng.
- *Trang tổng quan và trực quan hóa dữ liệu:* Hỗ trợ tích hợp cho hơn 100 thành phần widget. Thu thập và trực quan hóa dữ liệu từ các thiết bị và nội dung trong trang tổng quan bằng các tiện ích con.
- *Phép đo từ xa:* Phân tích phép đo từ xa đến và kích hoạt cảnh báo với xử lý sự kiện phức tạp. Hỗ trợ Api lưu trữ sự kiện để nắm bắt các chỉ số đo từ xa.

- *Hỗ trợ Rest API và RPC:* Quy trình làm việc dữ liệu có thể được thiết kế bằng cách sử dụng Rest API và RPC request.
- *Đẩy dữ liệu thiết bị:* Hỗ trợ đẩy dữ liệu thiết bị sang các hệ thống khác theo thời gian thực.
- *Tích hợp với các hàng đợi tin nhắn khác nhau:* Các trình kết nối khác nhau có sẵn để kết nối với các triển khai khác nhau của hàng đợi tin nhắn. Hỗ trợ triển khai nhiều hàng đợi tin nhắn: Kafka, RabbitMQ, AWS SQS, Azure Service Bus và Google Pub/Sub.
- *Hỗ trợ cơ sở dữ liệu kết hợp:* Lưu trữ tất cả các thực thể trong cơ sở dữ liệu SQL và dữ liệu đo từ xa trong cơ sở dữ liệu NoSQL.
- *Công cụ quy tắc:* Công cụ quy tắc tích hợp, để định cấu hình các quy tắc cho trạng thái thiết bị và các thông báo hoặc cảnh báo khác nhau có thể được tạo dựa trên phép đo từ xa của thiết bị. Các quy tắc xử lý dữ liệu có thể được thay đổi trong thời gian chạy dựa trên trạng thái thiết bị.
- *Chế độ triển khai, tiêu chuẩn và cụm:* Triển khai tại chỗ và đám mây được hỗ trợ cùng với chế độ tiêu chuẩn và cụm được hỗ trợ
- *Quản lý cảnh báo:* Tạo và quản lý các cảnh báo liên quan đến thiết bị, nội dung và khách hàng,... Chính sách quy tắc có thể được áp dụng để tăng cảnh báo khi thay đổi trạng thái thiết bị.

2.1.3 Phạm vi

Tập trung vào các API chức năng cơ bản trên Thingsboard:

- *Login endpoint*: chức năng này kiểm tra tính năng đăng nhập của hệ thống. Việc kiểm thử sẽ bao gồm xác thực tên đăng nhập và mật khẩu hợp lệ, xác thực tài khoản không hợp lệ, và xác thực thông tin sai.
- *Tenant controller*: chức năng này cho phép người dùng quản lý các đơn vị sử dụng hệ thống. Việc kiểm thử sẽ bao gồm kiểm tra tính năng tạo, cập nhật và xóa các đơn vị sử dụng hệ thống.

- *Customer controller*: chức năng này cho phép người dùng quản lý thông tin khách hàng. Việc kiểm thử sẽ bao gồm kiểm tra tính năng tạo, cập nhật và xóa thông tin khách hàng.
- *Manage dashboard*: chức năng này cho phép người dùng quản lý các bảng điều khiển. Việc kiểm thử sẽ bao gồm kiểm tra tính năng tạo bảng điều khiển, cập nhật thông tin bảng điều khiển và xóa bảng điều khiển.
- *Auth controller*: chức năng này kiểm soát việc xác thực và ủy quyền truy cập cho các tài khoản người dùng. Việc kiểm thử sẽ bao gồm xác thực tài khoản hợp lệ và xác thực tài khoản không hợp lệ.
- *Manage devices*: chức năng này cho phép người dùng quản lý các thiết bị được đăng ký trong hệ thống. Việc kiểm thử sẽ bao gồm kiểm tra tính năng tạo, cập nhật và xóa các thiết bị.
- *Control device*: chức năng này cho phép người dùng điều khiển các thiết bị trong hệ thống. Việc kiểm thử sẽ bao gồm kiểm tra tính năng bật/tắt thiết bị, cập nhật thông tin thiết bị và xóa thiết bị.
- *Manage telemetry*: chức năng này cho phép người dùng quản lý dữ liệu từ các thiết bị. Việc kiểm thử sẽ bao gồm kiểm tra tính năng lấy dữ liệu, cập nhật dữ liệu và xóa dữ liệu.

2.1.4 Những người sử dụng tài liệu này

Tài liệu kiểm thử này dành cho các nhà phát triển và nhà quản trị hệ thống, giúp họ hiểu rõ hơn về quy trình kiểm thử, có được kiến thức và kỹ năng cần thiết để kiểm thử các API trên Thingsboard. Bên cạnh đó, tài liệu này cũng có thể hữu ích cho các nhà thiết kế và kiểm thử ứng dụng IoT, giúp họ hiểu rõ hơn về việc sử dụng các API trên Thingsboard và cách kiểm tra tính đúng đắn và chất lượng của chúng.

2.2 Lịch trình công việc

Lịch trình công việc xây dựng kế hoạch kiểm thử API của Thingsboard bao gồm các bước chính sau:

- *Phân tích yêu cầu*: Để hiểu rõ yêu cầu của khách hàng và đảm bảo rằng các yêu cầu được đưa ra đều được đáp ứng.
- *Xác định các phương pháp kiểm thử*: Để đảm bảo rằng tất cả các khía cạnh của API đều được kiểm tra và chắc chắn rằng các kiểm thử sẽ được thực hiện một cách hiệu quả.
- *Chuẩn bị tài nguyên*: Để đảm bảo rằng các tài nguyên cần thiết để thực hiện các kiểm thử đều được sẵn sàng, ví dụ như cài đặt các công cụ kiểm thử, chuẩn bị dữ liệu kiểm thử, và môi trường kiểm thử.
- *Xác định kế hoạch kiểm thử*: Để đảm bảo rằng tất cả các phương pháp kiểm thử được đưa ra đều được thực hiện và đáp ứng yêu cầu của khách hàng.
- *Thực hiện kiểm thử*: Thực hiện các kiểm thử theo kế hoạch đã đưa ra, thu thập và phân tích kết quả kiểm thử.
- *Đánh giá và báo cáo*: Để đảm bảo rằng tất cả các phát hiện lỗi đều được ghi nhận và báo cáo, để có thể đưa ra các giải pháp khắc phục và cải tiến sản phẩm.
- *Tối ưu hóa kiểm thử*: Để đảm bảo rằng các kiểm thử sẽ được cải tiến liên tục, tối ưu hóa cho các phiên bản sản phẩm tiếp theo.

Việc lên lịch trình và thực hiện các bước trên sẽ giúp đảm bảo kế hoạch kiểm thử API của Thingsboard được thực hiện một cách hiệu quả, giảm thiểu sai sót và giúp sản phẩm được cải tiến liên tục. Dưới đây là lịch trình công việc cụ thể của nhóm:

Bảng 2. 1: Lịch trình công việc của nhóm

Milestone	Deliverables	Duration	Start Date	End Date	Người thực hiện
Lập kế hoạch kiểm thử	Tài liệu Test Plan	1 ngày	07/03/2023	08/03/2023	Tạ Thị Thơm
Thiết kế các Testcase	Tài liệu Test Plan	5 ngày	08/03/2023	13/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm

Xem lại các Testcase	Tài liệu Test Case	1 ngày	13/03/2023	14/03/2023	Tạ Thị Thơm
Thực thi các Testcase	Tài liệu Test case	7 ngày	14/03/2023	21/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm
Đánh giá kết quả kiểm thử	Tài liệu Test case	1 ngày	21/03/2023	22/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm
Viết báo cáo kiểm thử	Tài liệu Test case	4 ngày	22/03/2023	26/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm

2.3 Những yêu cầu về tài nguyên

Để xây dựng kế hoạch kiểm thử API cho ThingsBoard, cần đảm bảo rằng có đủ tài nguyên phần cứng và phần mềm, các công cụ kiểm thử cần thiết và môi trường kiểm thử phù hợp để thực hiện kiểm thử.

2.3.1 Phần cứng

Cần có máy tính (máy chủ) với đủ tài nguyên để chạy các kịch bản kiểm thử API. Điều này bao gồm bộ vi xử lý, bộ nhớ RAM, ổ cứng và card mạng đủ mạnh để xử lý các yêu cầu kiểm thử một các hiệu quả.

2.3.2 Phần mềm

Cần cài đặt các phần mềm cần thiết để thực hiện các kịch bản kiểm thử API bao gồm:

- Cài đặt hệ điều hành, trình duyệt web, công cụ Postman kiểm thử API và các thư viện hỗ trợ cho ngôn ngữ lập trình được sử dụng để tạo các kịch bản kiểm thử.

❖ Cài đặt Thingsboard để xây dựng và thực thi các kịch bản kiểm thử API trên môi trường của mình. Để cài đặt Thingsboard, cần thực hiện các bước sau:

- Tải phiên bản Thingsboard phù hợp từ trang web của Thingsboard
- Cài đặt phần mềm và thư viện phụ thuộc cần thiết để chạy Thingsboard
- Giải nén các tệp nén của Thingsboard và di chuyển nó đến vị trí muốn cài đặt
- Cấu hình Thingsboard bằng cách chỉnh sửa các tệp cấu hình hoặc sử dụng trình đơn cấu hình của Thingsboard
- Khởi động Thingsboard và kiểm tra xem nó đã chạy thành công chưa.

❖ Ngoài ra, có thể sử dụng ThingsBoard trên ThingsBoard.io nếu không thể cài đặt ThingsBoard trên máy tính. ThingsBoard.io cung cấp dịch vụ đám mây ThingsBoard và cho phép tạo các thiết bị ảo và bảng điều khiển để quản lý dữ liệu IoT.

Lưu ý: Ubuntu là một lựa chọn phổ biến cho việc cài đặt ThingsBoard. Bởi vì ThingsBoard được phát triển bằng Java và Ubuntu là một hệ điều hành Linux được phổ biến trong cộng đồng phát triển phần mềm. Ngoài ra, Ubuntu cũng hỗ trợ các công cụ quản lý gói phần mềm, máy chủ và được coi là một nền tảng ổn định hơn so với Windows.

2.3.3 Công cụ kiểm thử

Công cụ được nhóm sử dụng để kiểm thử đó là Postman và Swagger UI.

❖ Sử dụng công cụ Postman để kiểm thử API, để cài đặt công cụ này cần thực hiện các bước sau:

- Tải và cài đặt Postman trên máy tính từ trang chủ của Postman: https://www.postman.com/downloads/
- Tạo một tài khoản Postman để lưu trữ các yêu cầu và môi trường kiểm thử.
- Đăng nhập vào tài khoản Postman trên ứng dụng Postman trên máy tính.
- Tạo một môi trường để lưu trữ các biến môi trường. Biến môi trường là các giá trị động được sử dụng để xác định các yêu cầu API (ví dụ: URL, mã xác thực, tham số yêu cầu).

- Thêm các yêu cầu API vào Postman. Có thể thêm các yêu cầu API bằng cách tạo mới hoặc nhập từ tệp hoặc URL.
- Thực hiện kiểm thử API bằng cách chạy các yêu cầu API trong Postman và xem kết quả trả về.
- Lưu trữ các yêu cầu và môi trường kiểm thử trên tài khoản Postman để sử dụng lại trong các lần kiểm thử tiếp theo.

❖ Tài liệu API: Swagger là một công cụ giúp tạo tài liệu cho các RESTful API. Nó có thể giúp định nghĩa và tạo nên các yêu cầu API cho Thingsboard dễ dàng hơn, cũng như cung cấp tài liệu rất hữu ích.

Khi đã cài đặt Thingsboard thành công, Swagger được tích hợp sẵn trong Thingsboard. Để truy cập Swagger, cần truy cập vào giao diện quản trị của Thingsboard và thực hiện các bước sau:

- Đăng nhập vào giao diện quản trị của Thingsboard với tư cách người dùng có quyền truy cập API hoặc tài khoản quản trị hệ thống.
- Nhấp vào biểu tượng menu ở góc trên bên trái của giao diện quản trị.
- Chọn "API" từ menu và sau đó chọn "Swagger".
- Điều này sẽ mở trình duyệt web và hiển thị trang Swagger UI, cho phép khám phá và thực hiện các yêu cầu API của Thingsboard.

Lưu ý: Cần có quyền truy cập API hoặc quyền quản trị hệ thống để có thể truy cập Swagger trong Thingsboard. Nếu không thể truy cập Swagger, hãy kiểm tra xem tài khoản có quyền truy cập API hay không. Hoặc truy cập : http://YOUR_HOST:PORT/swagger-ui.html với HOST và PORT tương ứng.

2.3.4 Môi trường kiểm thử

Để kiểm thử API trên Thingsboard bằng POSTMAN, cần chuẩn bị các tài nguyên môi trường kiểm thử API như sau:

❖ *URL của Thingsboard server*: URL này phải bao gồm đầy đủ thông tin để POSTMAN có thể truy cập vào server Thingsboard. Ví dụ: "http://localhost:8080/api/v1" hoặc "https://example.thingsboard.io/api/v1"

- *Tên người dùng và mật khẩu*: để truy cập vào Thingsboard server, cần có tên người dùng và mật khẩu hợp lệ. Thông tin này cần được cung cấp trong các yêu cầu API.
- *Token truy cập*: Token truy cập được cung cấp bởi Thingsboard server và được sử dụng để xác thực yêu cầu API. Có thể tạo một token truy cập mới trong Thingsboard bằng cách đăng nhập vào tài khoản và chọn "API Keys" trong menu bên trái.
- *Biến môi trường*: POSTMAN cho phép sử dụng các biến môi trường để lưu trữ các giá trị động (ví dụ: URL, tên người dùng, token truy cập). Điều này giúp dễ dàng sử dụng lại các giá trị này trong các yêu cầu API khác nhau.
- *Các yêu cầu API*: Cần tạo các yêu cầu API cho các chức năng cụ thể muốn kiểm thử trên Thingsboard. Các yêu cầu API này cần phù hợp với định dạng yêu cầu API được Thingsboard yêu cầu.

Với các tài nguyên môi trường kiểm thử API trên, có thể sử dụng POSTMAN để kiểm thử API trên Thingsboard và đảm bảo tính ổn định của hệ thống.

2.4 Nhân sự

Kỹ sư kiểm thử API (API Test engineers): Đây là những người chịu trách nhiệm thiết kế và triển khai các kịch bản kiểm thử API bằng Postman. Các kỹ sư kiểm thử API phải có kiến thức chuyên môn về các giao thức web, các phương thức HTTP và cách sử dụng Postman.

Quản lý kiểm thử (Test managers): Đây là những người chịu trách nhiệm quản lý và giám sát quá trình kiểm thử API bằng Postman, đảm bảo rằng kế hoạch kiểm thử được triển khai đúng thời gian và ngân sách.

Nhân viên kiểm thử (Testers): Những người này chịu trách nhiệm thực hiện các ca kiểm thử API bằng Postman, báo cáo lỗi và đóng góp ý kiến để cải tiến sản phẩm.

2.5 Phạm vi kiểm thử

2.5.1 Những chức năng được kiểm thử

Tập trung vào kiểm thử các chức năng cơ bản của Thingsboard, bao gồm:

- Quản lý thiết bị: API để tạo, cập nhật và xóa thiết bị trên Thingsboard.

- ❖ Quản lý dữ liệu: API để lấy dữ liệu từ Thingsboard và thêm dữ liệu mới vào Thingsboard.
- ❖ Quản lý người dùng: API để tạo, cập nhật và xóa người dùng trên Thingsboard.
- ❖ Quản lý bảng điều khiển: API để tạo, cập nhật và xóa bảng điều khiển trên Thingsboard

2.5.2 Những chức năng không được kiểm thử

Kế hoạch kiểm thử API sẽ không tập trung vào kiểm thử các chức năng nâng cao hoặc phức tạp hơn của Thingsboard, bao gồm:

- ❖ Các tính năng liên quan đến tích hợp và mở rộng Thingsboard.
- ❖ Cài đặt và quản lý các luật (rule) và định tuyến (routing)
- ❖ Tích hợp với các hệ thống khác như Google Cloud, AWS, Azure
- ❖ Các tính năng liên quan đến bảo mật và phân quyền.

2.6 Chiến lược kiểm thử

Một trong những chiến lược kiểm thử tự động phổ biến được sử dụng trong kiểm thử API là chiến lược kiểm thử bằng kịch bản hoặc còn gọi là kiểm thử hướng kịch bản (Scenario-Based Testing).

Chiến lược kiểm thử bằng kịch bản tập trung vào việc tạo ra các kịch bản kiểm thử (test scenarios) dựa trên các tình huống sử dụng thực tế. Các kịch bản kiểm thử này có thể bao gồm các bước sử dụng API, các trường hợp sử dụng khác nhau, các giá trị đầu vào và kết quả mong đợi.

Với việc sử dụng Postman trên ThingsBoard, chúng ta có thể xây dựng các kịch bản kiểm thử bằng cách sử dụng tính năng Collection Runner của Postman. Các kịch bản kiểm thử có thể được tạo ra bằng cách sử dụng các API request được xác định trước và thiết lập các tham số và giá trị đầu vào khác nhau để thử nghiệm API trong các trường hợp sử dụng khác nhau.

Để sử dụng chiến lược kiểm thử bằng kịch bản trong Postman trên ThingsBoard, các bước cơ bản có thể là:

- ❖ Xác định các yêu cầu và trường hợp kiểm thử cần thiết để kiểm thử API trên ThingsBoard.
- ❖ Tạo các kịch bản kiểm thử bằng cách sử dụng Collection Runner của Postman.
- ❖ Cấu hình các tham số và giá trị đầu vào khác nhau để thử nghiệm API trong các trường hợp sử dụng khác nhau.
- ❖ Thực hiện các kịch bản kiểm thử bằng cách chạy Collection Runner.
- ❖ Kiểm tra và phân tích kết quả kiểm thử để đánh giá tính năng và độ tin cậy của API trên ThingsBoard.
- ❖ Lập báo cáo kết quả kiểm thử để báo cáo cho nhóm phát triển và quản lý dự án.

Chiến lược kiểm thử bằng kịch bản là một phương pháp kiểm thử rất hiệu quả và đảm bảo tính toàn diện của việc kiểm thử API trên ThingsBoard. Nó giúp giảm thiểu thời gian và chi phí cho việc kiểm thử và cải thiện độ tin cậy của hệ thống.

2.7 Điều kiện chấp nhận

Điều kiện chấp nhận để kiểm thử API của Thingsboard bao gồm các yêu cầu sau:

- ❖ Đã cài đặt thành công Thingsboard và có thể truy cập vào API của nó.
- ❖ Đã chuẩn bị một tài khoản người dùng với đầy đủ quyền truy cập API hoặc tài khoản quản trị hệ thống.
- ❖ Đã chuẩn bị các yêu cầu kiểm thử API cần thiết để kiểm tra tính năng của các chức năng API của Thingsboard.
- ❖ Đã chuẩn bị các tài nguyên phần mềm và phần cứng cần thiết để chạy các bộ kiểm thử API.
- ❖ Đã thiết lập môi trường kiểm thử tương tự với môi trường sản phẩm để đảm bảo tính chính xác và khả năng tái sử dụng của các bộ kiểm thử API.
- ❖ Đã đảm bảo tính bảo mật cho các tài khoản và thông tin đăng nhập được sử dụng trong quá trình kiểm thử API.

Các yêu cầu này có thể được điều chỉnh tùy thuộc vào mục đích kiểm thử và yêu cầu của dự án. Tuy nhiên, việc đảm bảo tính chính xác, khả năng tái sử dụng và tính bảo

mật của các bộ kiểm thử API là rất quan trọng để đảm bảo tính ổn định và đáng tin cậy của sản phẩm.

2.8 Bảng phân công công việc

Bảng 2. 2: Bảng phân công công việc

Nhiệm vụ	Thời gian	Người thực hiện
Lập kế hoạch kiểm thử	07/03/2023	Tạ Thị Thơm
Thiết kế các Testcase	08/03/2023 đến 13/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm
Xem lại các Testcase	14/03/2023	Tạ Thị Thơm
Thực thi các Testcase	14/03/2023 đến 21/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm
Đánh giá kết quả kiểm thử	22/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm
Viết báo cáo kiểm thử	23/03/2023 đến 26/03/2023	Hoàng Thị Hường Nguyễn Thị Nhị Tạ Thị Thơm

2.9 Defect tracking

2.9.1 Phân loại lỗi

Defect tracking (theo dõi lỗi) là một quá trình quản lý, theo dõi và giải quyết các lỗi trong phần mềm. Đây là một phần quan trọng của quản lý chất lượng phần mềm và được sử dụng để đảm bảo rằng các lỗi trong phần mềm được phát hiện và giải quyết một cách hiệu quả. Các lỗi được ghi lại và theo dõi thông qua một hệ thống quản lý lỗi và quá trình theo dõi này bao gồm các bước như phát hiện lỗi, phân loại lỗi, ưu tiên hóa lỗi, gán người xử lý, sửa lỗi, kiểm tra lại và đóng lỗi. Quá trình theo dõi lỗi giúp cho những lỗi phát hiện được sớm và được giải quyết kịp thời trước khi tác động đến chất lượng sản phẩm phần mềm và đảm bảo sự ổn định và tin cậy của phần mềm.

Lỗi (bug) là sự cố hoặc sai sót trong phần mềm, gây ra kết quả không mong muốn hoặc không hoạt động đúng cách. Để quản lý lỗi trong phần mềm, các lỗi phải được phân loại theo một số tiêu chí khác nhau, ví dụ như:

- *Mức độ ưu tiên*: các lỗi có mức độ ảnh hưởng nghiêm trọng đến chức năng của phần mềm sẽ được ưu tiên xử lý trước.
- *Loại lỗi:* lỗi cú pháp, lỗi logic, lỗi hệ thống, lỗi giao diện người dùng,...
- *Vị trí của lỗi:* các lỗi được phân loại theo vị trí của chúng trong phần mềm, chẳng hạn như lỗi ở cơ sở dữ liệu, lỗi ở phần mềm máy chủ, lỗi ở phần mềm client,...
- *Trạng thái:* lỗi đã được xác nhận, lỗi đã được sửa chữa, lỗi đã được kiểm tra lại,...

2.9.2 Quy trình xử lý lỗi

Quy trình xử lý lỗi là một quy trình quản lý chất lượng phần mềm giúp đảm bảo rằng các lỗi được xác định, phân loại và xử lý đầy đủ. Một số bước cơ bản trong quy trình xử lý lỗi bao gồm:

- *Ghi nhận lỗi:* Tìm kiếm các lỗi trong phần mềm bằng các kỹ thuật kiểm thử phần mềm.
- *Phân loại lỗi*: Phân loại lỗi theo các tiêu chí như đã nêu ở trên.
- *Xác định ưu tiên*: Xác định độ ưu tiên của các lỗi để quyết định xử lý trước những lỗi nào.
- *Gán người xử lý:* Gán các lỗi cho nhân viên có năng lực và kỹ năng để xử lý chúng.
- *Sửa chữa lỗi:* Các lỗi được xác định, phân loại, ưu tiên và gán người xử lý sẽ được sửa chữa.
- *Kiểm tra lại*: Sau khi sửa chữa lỗi, kiểm tra lại phần mềm để đảm bảo rằng các lỗi đã được sửa chữa một cách đầy đủ.
- *Đóng lỗi*: Khi các lỗi đã được sửa chữa đầy đủ, chúng sẽ được đóng lại và ghi lại trong hệ thống quản lý lỗi.

Khi test API trên Postman, có một số công cụ defect tracking có thể được sử dụng để quản lý các lỗi phát hiện được trong quá trình kiểm thử. Dưới đây là một số công cụ phổ biến và được nhóm sử dụng:

- *Postman Collection Runner:* Cho phép chạy nhiều bộ kiểm thử đồng thời và tự động ghi lại các kết quả kiểm thử để theo dõi.
- *Postman Test Results:* Cho phép theo dõi các kết quả kiểm thử bằng cách lưu trữ các báo cáo chi tiết về các kết quả kiểm thử.
- *JIRA:* Là một công cụ quản lý dự án và defect tracking được sử dụng rộng rãi, có thể được tích hợp với Postman để quản lý các lỗi được phát hiện trong quá trình kiểm thử.

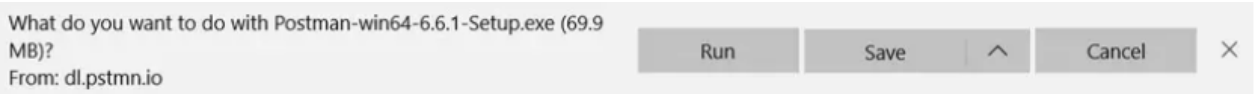
CHƯƠNG 3: TIẾN HÀNH CÀI ĐẶT VÀ DEMO THỰC NGHIỆM VỚI CÔNG CỤ POSTMAN

3.1 Cài đặt và cấu hình

3.1.1 Cách cài đặt

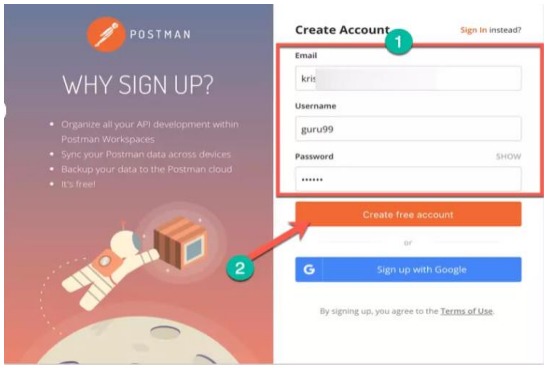
Bước 1: Truy cập https://www.postman.com/downloads/ và chọn nền tảng mong muốn trong số Mac, Windows hoặc Linux. Nhấp vào Download the App.

Bước 2: Tải xuống đang trong tiến trình tải sẽ hiển thị trên trang web. Khi quá trình tải xuống hoàn tất, nhấp vào Run.



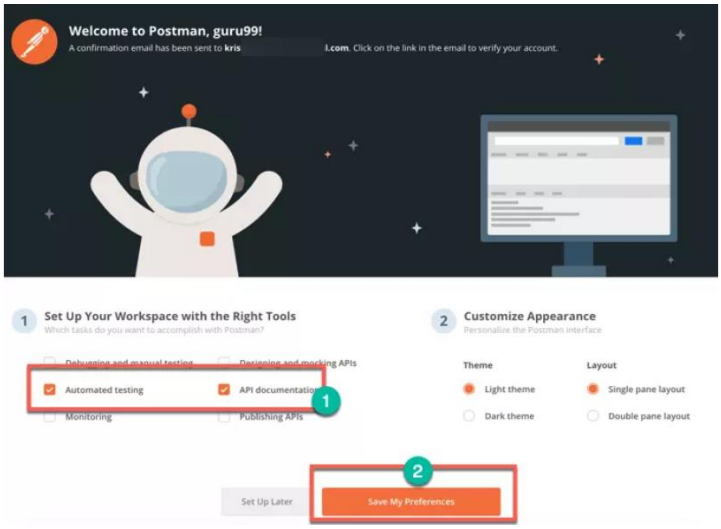
Bước 3: Bắt đầu cài đặt

Bước 4: Trong cửa sổ tiếp theo, đăng ký tài khoản postman

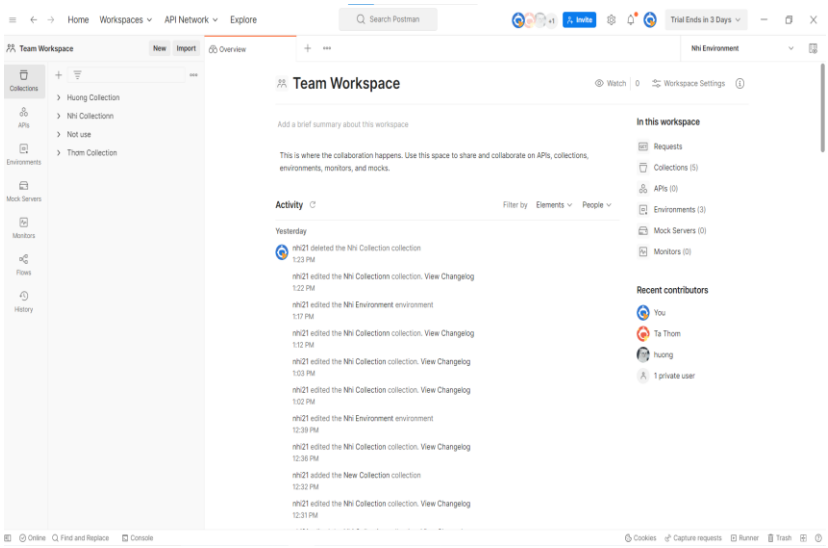


LƯU Ý: Có hai cách để đăng ký tài khoản Postman. Một là tạo tài khoản Postman riêng và hai là sử dụng tài khoản Google. Mặc dù Postman cho phép người dùng sử dụng công cụ mà không cần đăng nhập, đăng ký đảm bảo rằng collection của bạn được lưu và có thể được truy cập để sử dụng sau này.

Bước 5: Chọn các workspace tools bạn cần và nhấp vào lưu tùy chọn của tôi.



Bước 6: Bạn sẽ thấy màn hình bắt đầu.

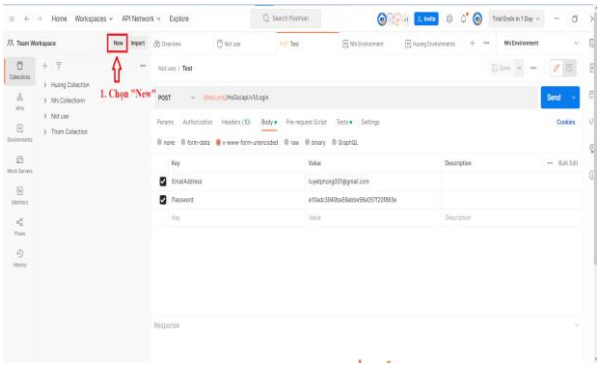


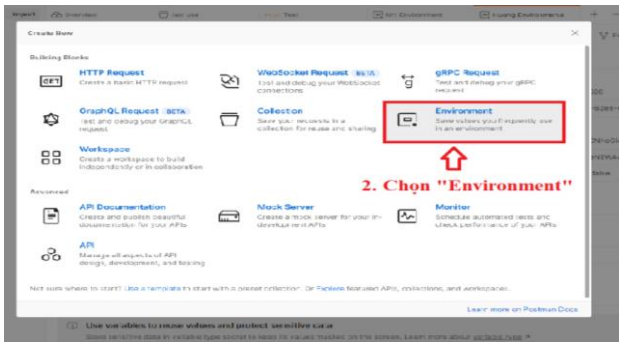
3.1.2 Cấu hình Postman

Postman là một công cụ giúp phát triển và kiểm thử các API . Khi sử dụng Postman, người dùng có thể thực hiện nhiều tác vụ khác nhau như gửi yêu cầu API, kiểm tra và xem phản hồi, quản lý các môi trường và biến, và nhiều hơn nữa. Để sử dụng Postman hiệu quả, người dùng cần cấu hình một số thiết lập. Dưới đây là một số cấu hình tiêu biểu:

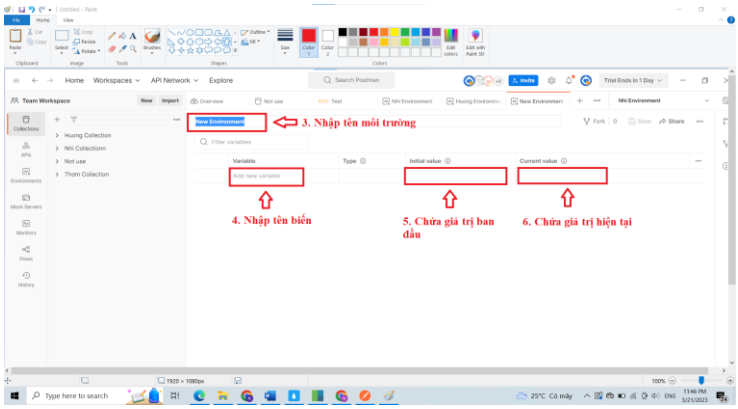
❖ Cấu hình môi trường và biến:

Bước 1: Mở Postman và tạo một môi trường mới bằng cách chọn biểu tượng "New" trên thanh bên trái như hình dưới đây.

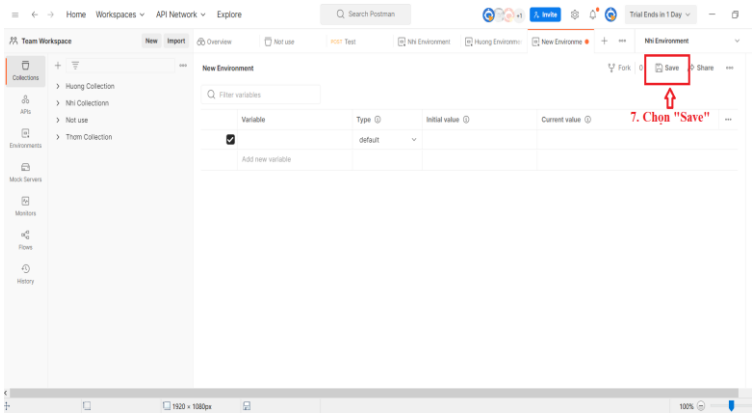




Bước 2: Nhập tên biến môi trường và các biến cần thiết vào Initial Value

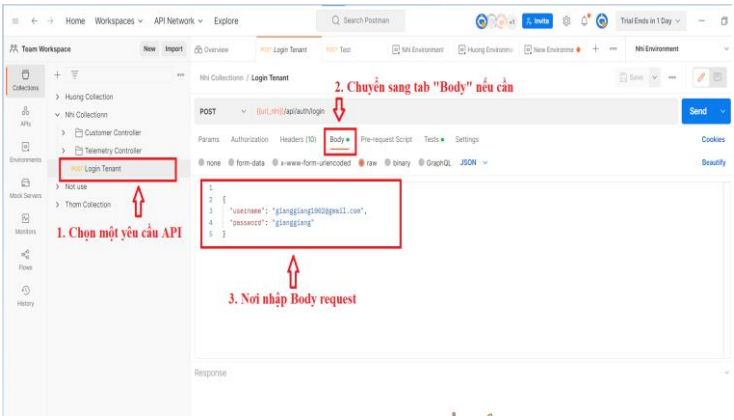


Bước 3: Chọn "Save" để lưu cấu hình.

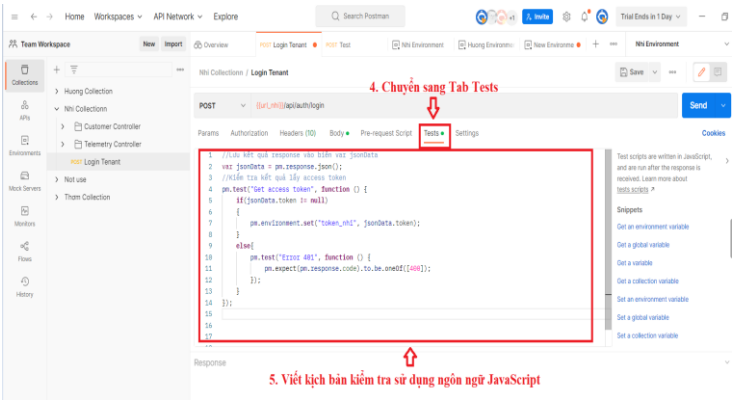


❖ Cấu hình test:

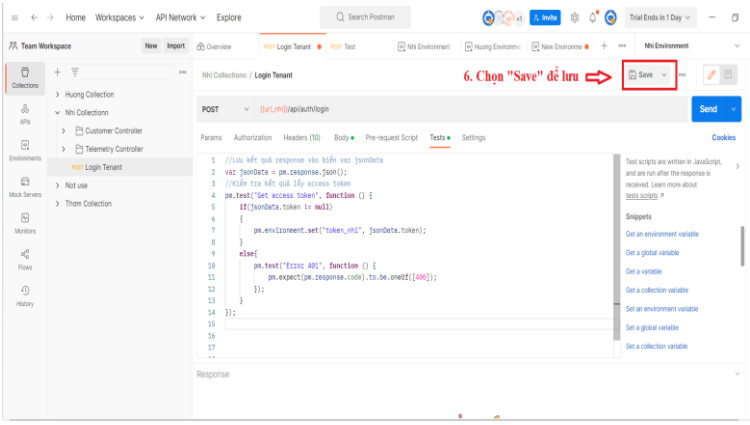
Bước 1: Chọn một yêu cầu API và chuyển sang tab “Body” nếu cần. Trong đó, Tab Body là nơi để xác định các nội dung (payload) được gửi đến server trong các yêu cầu HTTP, chẳng hạn như phương thức POST, PUT và PATCH. Đồng thời, cung cấp các tùy chọn để chỉ định kiểu dữ liệu của payload bao gồm các định dạng như: JSON, XML, HTML, TEXT, form data, .. Người dùng có thể tạo và chỉnh sửa nội dung payload trực tiếp trong Tab Body hoặc tải lên từ các tệp tin khác.



Bước 2: Chuyển sang Tab Tests để viết các kịch bản kiểm tra và sử dụng ngôn ngữ JavaScript.

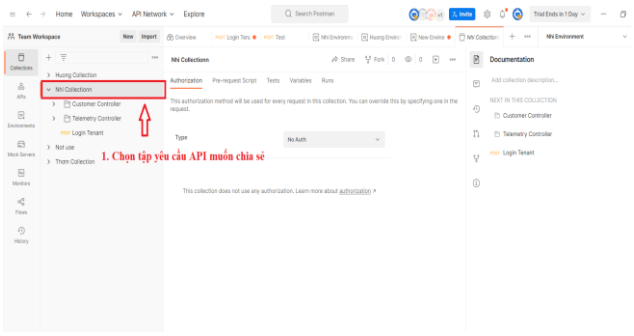


Bước 3: Lưu cấu hình bằng cách chọn “Save”.

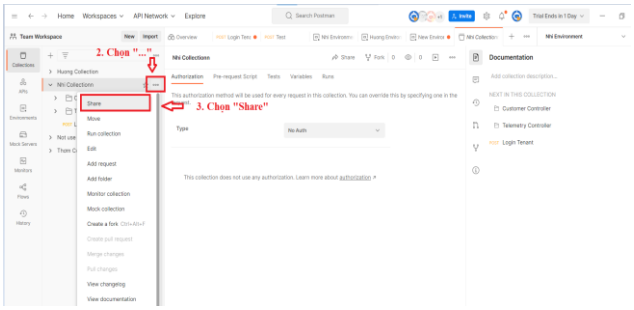


❖ Cấu hình chia sẻ:

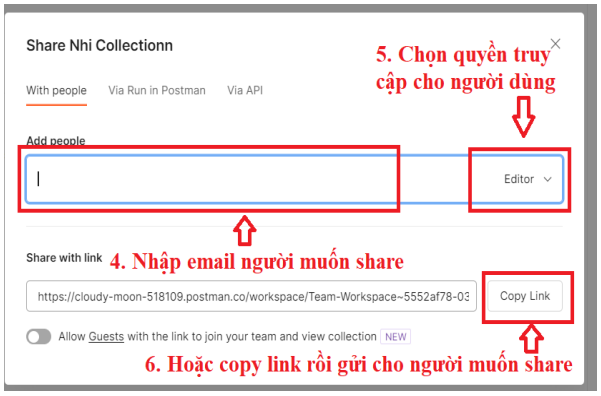
Bước 1: Chọn tập yêu cầu API muốn chia sẻ.



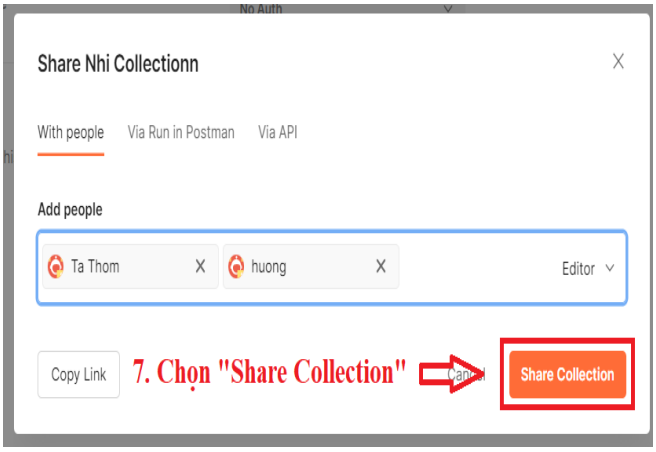
Bước 2: Chọn "Share" trên thanh bên trái.



Bước 3: Nhập email người muốn chia sẻ và chọn các quyền truy cập cho người dùng hoặc copy link rồi gửi link đó cho người muốn share.



Bước 4: Chọn "Share Collection" để gửi yêu cầu chia sẻ.



3.2 Cách viết một kịch bản với Postman

Để viết một kịch bản kiểm thử với Postman, làm theo các bước sau:

- Tạo một collection trong Postman để chứa tất cả các request liên quan đến kịch bản kiểm thử.
- Tạo một hoặc nhiều request trong collection để kiểm tra API. Các request có thể bao gồm các phương thức HTTP như GET, POST, PUT, DELETE và PATCH.
- Chọn tab "Tests" trên trang request và viết các kiểm tra để đảm bảo rằng API hoạt động đúng như mong muốn. Các kiểm tra có thể bao gồm việc kiểm tra mã trạng thái HTTP, nội dung của phản hồi và thời gian phản hồi.
- Tạo một environment trong Postman để lưu trữ các biến như URL của API và các thông số khác. Điều này sẽ giúp dễ dàng cập nhật các giá trị này khi chạy kịch bản kiểm thử trên các môi trường khác nhau.
- Thêm các biến môi trường vào các request bằng cách sử dụng cú pháp {{biến_môi_trường}}.
- Chạy kịch bản kiểm thử bằng cách nhấn vào nút "Run" trên Postman và chọn môi trường muốn chạy kịch bản trên.
- Kiểm tra kết quả kiểm thử và sửa các lỗi nếu cần thiết.

3.3 Xây dựng kịch bản test (Test Scenario) (kèm bản excel chi tiết)

❖ *Check Methods*:

1. Kiểm tra phương thức HTTP (GET, POST, PUT, DELETE) chính xác đang được sử dụng cho mỗi yêu cầu API.
2. Kiểm tra API hỗ trợ các phương thức HTTP dự kiến cho từng tài nguyên.
3. Kiểm tra API phản hồi chính xác với các biến thể trong phương thức HTTP.

❖ *Check Parameters*:

1. Kiểm tra tất cả các tham số bắt buộc đều có trong yêu cầu API.

2. Kiểm tra API trả về thông báo lỗi chính xác nếu bất kỳ tham số bắt buộc nào bị thiếu hoặc không hợp lệ.
3. Kiểm tra các tham số tùy chọn có thể được bỏ qua mà không gây ra lỗi.
4. Kiểm tra API hỗ trợ các tham số dự kiến cho từng tài nguyên.

❖ *Check Content-Type:*

1. Kiểm tra các tiêu đề phản hồi và yêu cầu API bao gồm đúng loại nội dung.
2. Kiểm tra API trả về thông báo lỗi chính xác nếu loại nội dung không chính xác hoặc không được hỗ trợ.

❖ *Check Data Parameters:*

1. Kiểm tra API xử lý chính xác các cấu trúc dữ liệu phức tạp, chẳng hạn như JSON hoặc XML.
2. Kiểm tra API trả về thông báo lỗi chính xác nếu bất kỳ tham số dữ liệu nào bị thiếu hoặc không hợp lệ.

❖ *Check Response Data:*

1. Kiểm tra API trả về dữ liệu dự kiến trong phản hồi.
2. Kiểm tra API trả về thông báo lỗi chính xác nếu không tìm thấy tài nguyên được yêu cầu.
3. Kiểm tra API trả về thông báo lỗi chính xác nếu có lỗi phía máy chủ.

❖ *Check Response Code/ Message:*

1. Kiểm tra API trả về mã phản hồi chính xác cho từng yêu cầu API.
2. Kiểm tra API trả về thông báo phản hồi chính xác cho từng mã phản hồi.

❖ *Check Error Response:*

1. Kiểm tra API trả về thông báo lỗi chính xác cho từng tình trạng lỗi.
2. Kiểm tra API trả về đúng mã lỗi cho từng tình trạng lỗi.
3. Kiểm tra phản hồi lỗi bao gồm tất cả các chi tiết liên quan về lỗi.

❖ *Check Authorization:*

1. Kiểm tra API yêu cầu ủy quyền cho các tài nguyên được bảo vệ.

2. Kiểm tra API trả về thông báo lỗi chính xác nếu ủy quyền không thành công hoặc không hợp lệ.

❖ *Check Limit Data:*

1. Xác minh API xử lý chính xác dữ liệu phân trang hoặc giới hạn.
2. Xác minh API trả về thông báo lỗi chính xác nếu dữ liệu giới hạn bị vượt quá hoặc không hợp lệ.

❖ *Check Data Not Exist :*

1. Xác minh API trả về phản hồi chính xác nếu không tồn tại dữ liệu.
2. Xác minh API trả về mã phản hồi và thông báo chính xác.

3.4 Thiết kế các test case và thực hiện test

3.4.1 Đặc tả API kiểm thử

❖ Các chức năng kiểm thử bao gồm:

- ✓ Login-endpoint
- ✓ Customer controller:
- ✓ Tenant controller
- ✓ Auth controller
- ✓ Control device
- ✓ Manage device
- ✓ Manage dashboard
- ✓ Manage telemetry

❖ Chi tiết đặc tả:

Bảng 3. 1: Đặc tả API kiểm thử

Nội dung	Chi tiết	Phương thức	API	Status code
Login endpoint	Login method used to authenticate user and get JWT token data	POST	/api/auth/login	200, 401
Control device	Send two way RPC request	POST	/api/plugins/rpc/twoway/{deviceId}	200, 400, 401, 500

Manage dashboard	Assign the Dashboard to specified Customer. Returns the Dashboard object.	POST, GET, DELETE	/api/customer/{customerId}/dashboard/{dashboardId}	200, 400, 401, 403, 500
Customer controller	Tạo customer	POST	/api/customer	200, 400, 404, 500
	Xem thông tin customer	GET	/api/customer/{id_customer }/shortInfo	
	Xoá customer	DELETE	/api/customer/{id_customer}	
Manage telemetry	Tạo devices attributes	POST	/api/plugins/telemetry/{deviceId}/SHARED_SCOPE	200, 400
	Chức năng: Lấy tất cả attribute keys trong SHARED_SCOPE	GET	/api/plugins/telemetry/{entityType}/{entityId}/keys/attributes/SHARED_SCOPE	
	Xóa Devices Attribute	DELETE	/api/plugins/telemetry/{devices_id}/SHARED_SCOPE	
Manage devices	Cập nhật thuộc tính cho thiết bị	POST	/api/v1/{deviceToken}/attributes	200, 400, 401, 500
	Lấy các thuộc tính cụ thể của thiết bị	GET	/api/v1/{deviceToken}/attributes{?clientKeys,sharedKeys}	
	Đăng dữ liệu chuỗi thời gian cho thiết bị.	POST	/api/v1/{deviceToken}/attributes{?clientKeys,sharedKeys}	
Auth controller	Thay đổi mật khẩu	POST	/api/auth/changePassword	200, 400, 401

	Email yêu cầu đặt lại mật khẩu;	POST	/api/noauth/resetPasswordByEmail	
Tenant controller	Tạo hoặc cập nhật Tenant	POST	/api/tenant	200, 400, 401, 404, 405, 500
	Lấy thông tin Tenant dựa trên Id được cung cấp.	GET	/api/tenant/{tenantId}	
	Xóa Tenant	DELETE	/api/tenant/{tenantId}	

3.4.2 Thiết kế các testcase

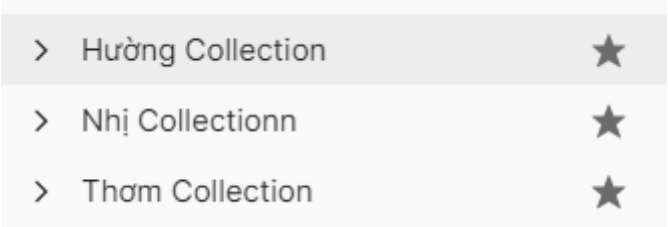
Bảng 3. 2: Thiết kế testcase

Tester	ID	Bản rõ	Tổng testcase
Thơm	LE	Login endpoint	6
Nhị	CC	Customer controller	41
Hường	TC	Tenant controller	33
Hường	AC	Auth controller	11
Thơm	CD	Control device	6
Hường	MD	Manage device	12
Thơm	MDb	Manage dashboard	11
Nhị	MT	Manage telemetry	10
Tổng kết			130

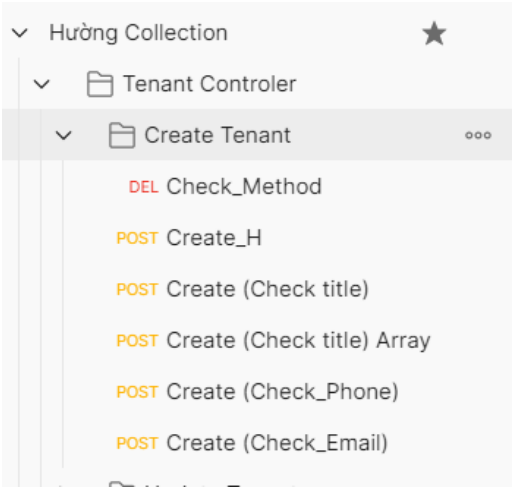
3.5 Kiểm thử tự động API với Postman trên Thingsboard

❖ Tạo collection chứa các request do từng người trong nhóm phụ trách bao gồm:

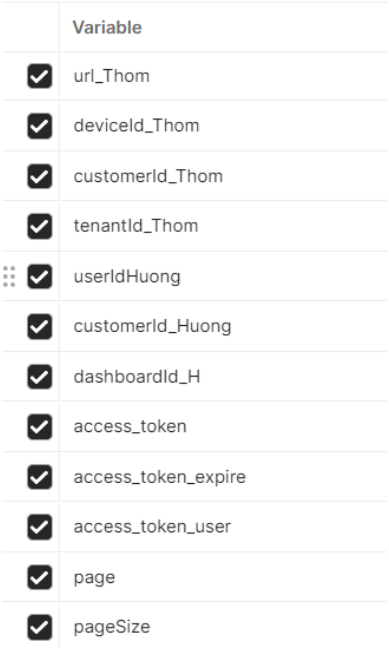
- Thơm collection
- Hường collection
- Nhị collection



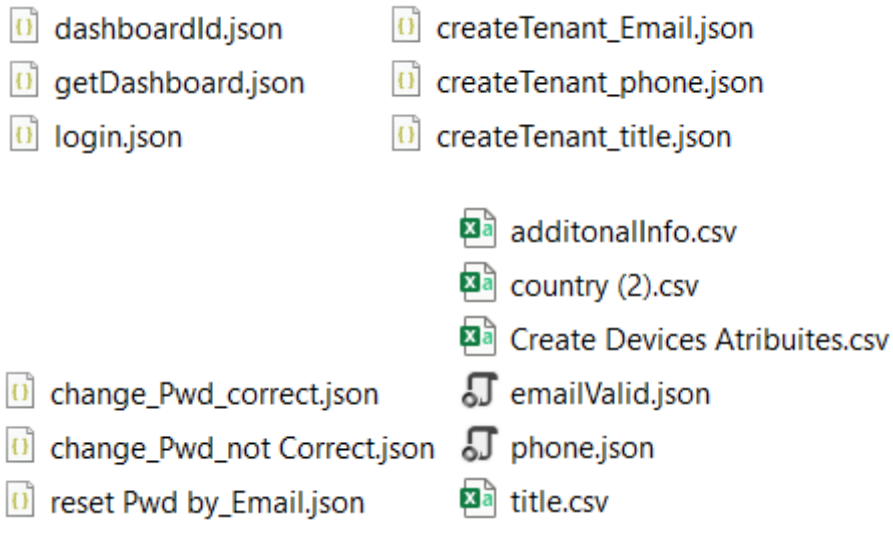
Bên trong là các request:



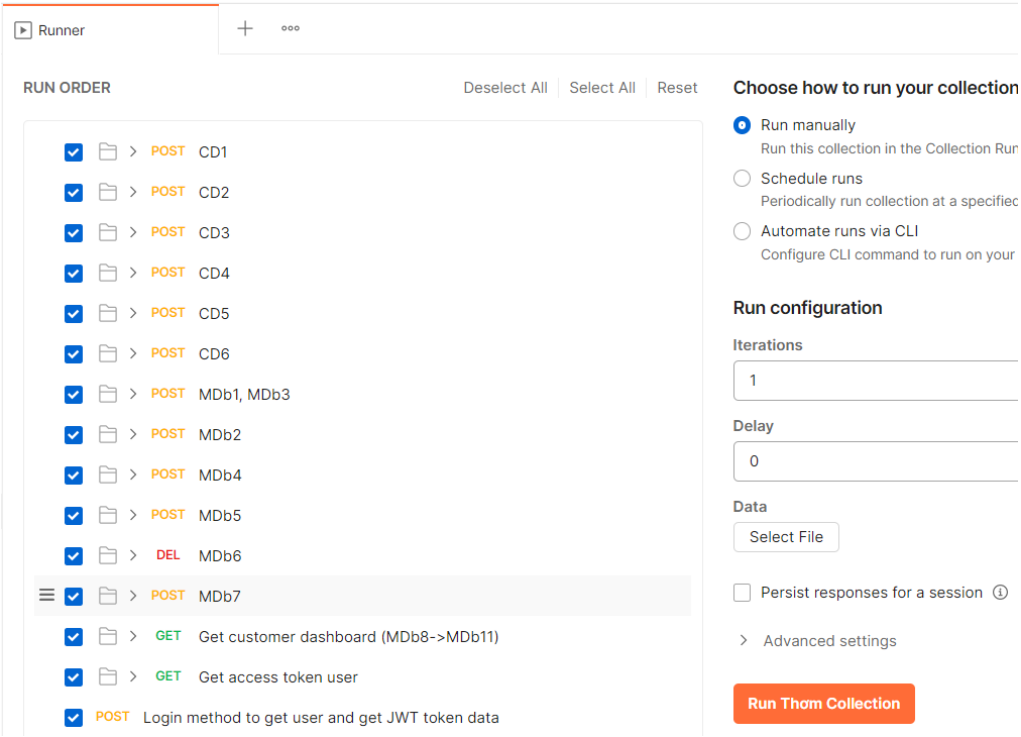
❖ Tạo các biến môi trường:



❖ Chuẩn bị các file dữ liệu được sử dụng trong quá trình test:



❖ Sử dụng chức năng collection runner để test tự động collection:



❖ Các câu lệnh script test được sử dụng:

- Thiết lập biến môi trường:

```
pm.environment.set("url_Thom", "http://192.168.20.108:8080");
```

- Kiểm tra mã code trả về có đúng theo tài liệu không:

```
pm.test("Get access token", function () {
    pm.environment.set("access_token", jsonData.token);
    pm.response.to.have.status(200);
});
```

- Phân tích và so sánh giá trị trong response trả về với kết quả kỳ vọng:

```
pm.test("Kiểm tra giá trị của số 5 trong chuỗi JSON", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData["5"]).to.equal(true); // Kiểm tra giá trị của
        hay không
});
```

- Kiểm tra độ dài mong muốn:

```
pm.test("Kiểm tra với email có độ dài lớn hơn 255 kí tự", function() {
    pm.expect(email.length).to.be.at.most(256);
});
```

- Kiểm tra mã code theo hàm expect:

```
pm.test("Error 401", function () {
    pm.expect(pm.response.code).to.be.oneOf([400]);
});
```

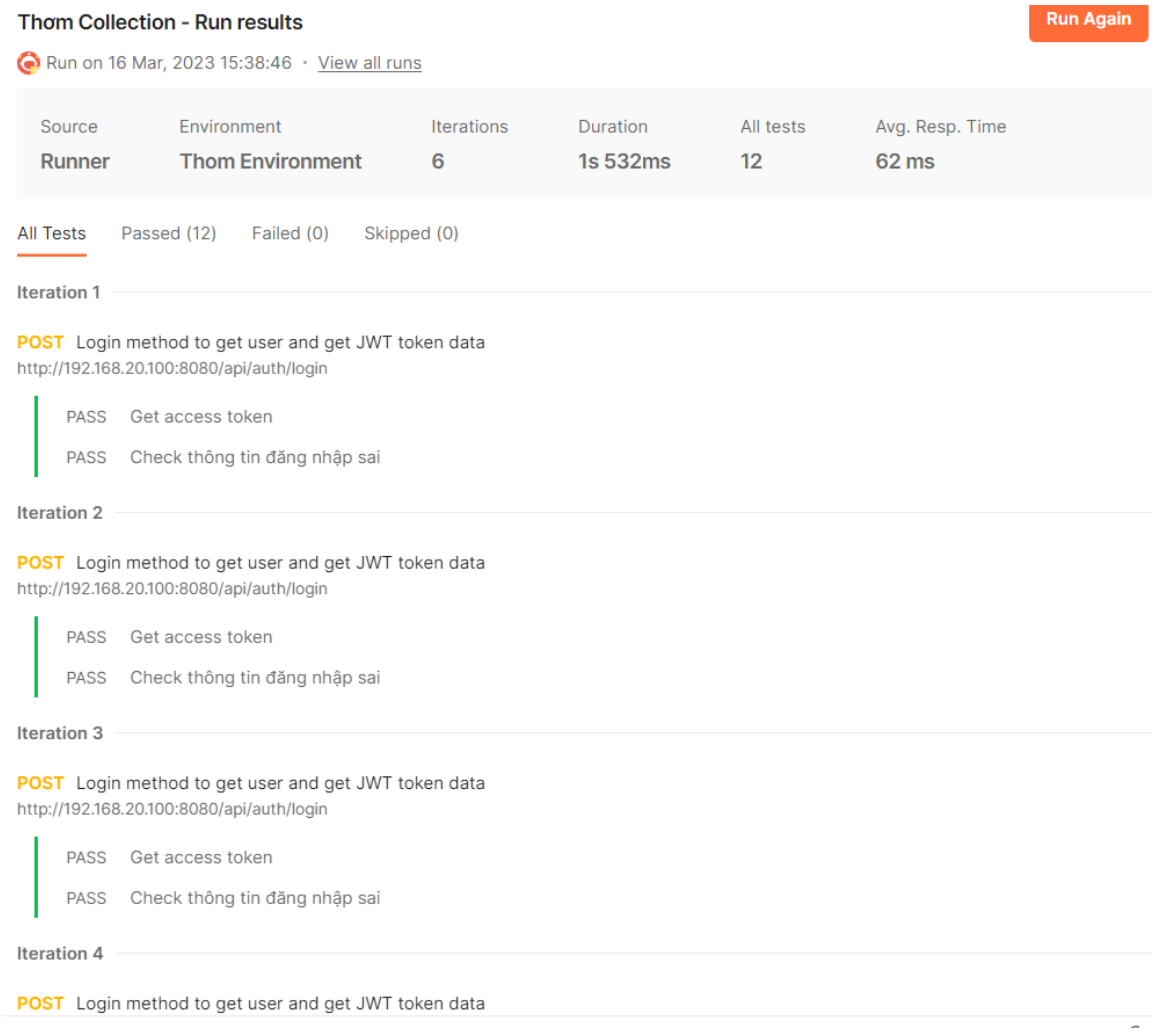
- Kiểm tra một request tới một API endpoint và mong đợi kết quả trả về là true:

```
pm.test("Kiểm tra với title là null ", function () {
  pm.expect(true).to.be.true;
});
```

3.6 Thực hiện test và phân tích lỗi

❖ Thực hiện test và kết quả test:

Kết quả trả về trên Postman sẽ có dạng:



Hình 3. 1: Kết quả trên Postman

Kết quả test:

Người phụ trách	ID	Bản rõ	Tổng Testcase	PASS	Tỉ lệ Pass	Tỉ lệ fail
THƠM	LE	Login endpoint	6	6	100,00%	0,00%
NHỊ	CC	Customer controller	41	35	85,37%	14,63%
HƯỜNG	TC	Tenant controller	33	29	87,88%	12,12%
HƯỜNG	AC	Auth controller	11	10	90,91%	9,09%
THƠM	CD	Control device two way	6	4	66,67%	33,33%
HƯỜNG	MD	Manage devices	12	12	100,00%	0,00%
THƠM	MDb	Manage Dashboard	11	11	100,00%	0,00%
NHỊ	MT	Manage telemetry	10	10	100,00%	0,00%
TỔNG KẾT			130	117	90,00%	10,00%

Hình 3. 2: Kết quả test

❖ Phân tích lỗi:

Lỗi được phân loại dựa trên ba mức độ: Mức High (mức cao), mức Medium (mức trung bình) và mức Low (mức thấp).

- Mức High: Lỗi ở cấp độ này là rất nghiêm trọng và làm sập hệ thống hoặc thay đổi chức năng. Tuy nhiên, một số phần của hệ thống vẫn hoạt động.
- Mức Medium: Lỗi ở cấp độ này gây ra một số hành vi không mong muốn, nhưng hệ thống vẫn hoạt động.
- Mức Low: Lỗi ở cấp độ này sẽ không gây ra bất kỳ sự cố lớn nào của hệ thống.

Mức độ lỗi	Số lượng
High	2
Medium	3
Low	8

❖ Chi tiết lỗi:

ID	Test Title	Request body	Expect results	Actual results	Error level

CC15	Tạo Customer với additionalInfo là kiểu số nguyên	{
		"title": "e",
		"additionalInfo": 1234
		}
			Không tạo Customer và trả về trạng thái 500	Tạo Customer thành công và trả về mã trạng thái 200	M

CC29	Tạo Customer với country là chuỗi ngẫu nhiên không phải một tên nước	{
		"title": "nc",
		"country": "!@#$%"
		}
			Không tạo được Customer	Tạo thành công trả về mã trạng thái 200	L

CC30	Tạo Customer với country là kiểu số nguyên	{
		"title": "nd",
		"country": 1234
		}
			Không tạo được Customer và trả về mã trạng thái 500	Tạo thành công trả về mã trạng thái 200	L

AC7	Đổi mật khẩu với mật khẩu mới không đủ mạnh (không chứa chữ hoa, chữ thường, số hoặc ký tự đặc biệt)	{
		"currentPassword": "Tenant@11",
		"newPassword": "11111111"
		}
			Mã trạng thái 400 Bad Request "Weak password"or "Invalid password", "Your new password must be at least 6 characters long and contain at least one uppercase letter, one lowercase letter, one number, and one special character."	Đổi mật khẩu thành công và phản hồi mã trạng thái 200	H

CD5	Định dạng sai kiểu boolean trong "enabled"
	{
	"2": true,
	"3": false,
	"4": false,
	"5": true,
	"6": false
	}
		{
		"method": "setGpioStatus",
		"params": {
		"pin": 5,
		"enabled": "@"
		},
		"timeout": 500
		}
			Return error về định dạng dữ liệu	Return 200 OK, thay đổi true->false
				{
				"2": true,
				"3": false,
				"4": false,
				"5": false,
				"6": false
				}
					H

CD6	Kiểm tra tính đúng đắn của kiểu dữ liệu
	{
	"2": true,
	"3": false,
	"4": false,
	"5": false,
	"6": false
	}
		{
		"method": "setGpioStatus",
		"params": {
		"pin": 3,
		"enabled": "ta là người xấu đây! haha"
		},
		"timeout": 500
		}
			Return error về định dạng của kiểu dữ liệu	Return 200 OK, Chuỗi bắt đầu bằng t là True, bằng f là False
				{
				"2": true,
				"3": true,
				"4": false,
				"5": false,
				"6": false
				}
					L

ID	Test Title	Request body	Expect results	Actual results	Error level

TC4	Tạo Tenant với trường bắt buộc title chỉ có khoảng trắng	{
		"title": " "
		}
			Return error 400, "Tenant title should be specified!"	Tạo Tenant thành công, trả về mã trạng thái 200	L

TC7	Tạo Tenant trùng với title Tenant đã tồn tại	{
		"title": "Alam"
		}
			"status": 400, "Cannot create a new tenant with a title that already exists. Please enter a different name."	Tạo Tenant thành công và trả về mã trạng thái 200	L

TC13	Tạo Tenant có số điện thoại quá ngắn	{
		"title": "Silo A2",
		"email": "A2@company.com",
		"phone": "+84 1"
		}
			"status": 400, "message": "Invalid phone format '+84 1'!"	Tạo Tenant thành công và trả về mã trạng thái 200	L

TC14	Tạo Tenant có định dạng số điện thoại chứa ký tự đặc biệt không cho phép ở giữa số	{ "title": "CTY A ", "phone": "+84 35773@$6634" }	"status": 400, "message": "Invalid phone format '+84 35773@$6634'!"	Tạo Tenant thành công và trả về mã trạng thái 200	L

CC9	Tạo Customer với title là khoảng trắng	{
		"title": " "
		}
			Không tạo Customer và trả về mã trạng thái 400	Tạo Customer thành công và trả về mã trạng thái 200	L

CC13	Tạo Customer với additionalInfo là kiểu string	{
		"title": "c",
		"additionalInfo": "abc"
		}
			Không tạo Customer và trả về trạng thái 500	Tạo Customer thành công và trả về mã trạng thái 200	M

CC14	Tạo Customer với additionalInfo là kiểu mảng	{
		"title": "d",
		"additionalInfo": [{ "a": 1 },
		{ "b": 3 }]
		}
			Không tạo Customer và trả về trạng thái 500	Tạo Customer thành công và trả về mã trạng thái 200	M

3.7 Đánh giá kết quả

❖ Đánh giá:

- Tất cả các test case đã được thực thi.
- Tỷ lệ Test Case đạt khá cao : 90%.
- Tất cả các lỗi tìm thấy đều được rà soát lại nhiều lần và ghi nhận lý do rõ ràng.

❖ Kết quả đạt được:

- Hiểu được cách hoạt động và làm việc của công cụ Postman và API.
- Thiết kế và thực thi các testcase.
- Quan sát và tìm ra lỗi, đánh giá được lỗi theo mức độ
- Viết ra được tài liệu kiểm thử testcase.
- Giúp áp dụng được vào các dự án thực tế.

TÀI LIỆU THAM KHẢO

[1]. Nguyễn Văn Vỵ - Nguyễn Việt Hà, Giáo trình Kỹ nghệ phần mềm, Nhà xuất bản Giáo dục Việt Nam, 2009

[2]. Giang Nguyễn, API Testing với Postman, file PDF xuất bản năm 2018.

[3]. https://zapier.com/learn/apis/

[4]. https://learning.postman.com/docs/introduction/overview/

[5]. ThS. Thái Thị Thanh Vân, slide bài giảng, 2022.