LUMENDATA

Data Sheet

REST API Automation using POSTMAN

Step-by-Step Guide

5201 GREAT AMERICAN PARKWAY, SUITE 320
SANTA CLARA, CA 95054
Tel: (855) 695-8636
E-mail: info@lumendata.com
Website: www.lumendata.com

Introduction

Postman is a powerful tool for performing integration testing with your API. It allows for repeatable, reliable tests that can be automated and used in a variety of environments and includes useful tools for persisting with data and simulating how a user might be interacting with the system. **This document describes how to automate POSTMAN Collection for API testing and how POSTMAN is one of the best tools for API automation.**

Postman Automation

Postman allows user to automate test cases in JavaScript with salient features like writing test suites, building requests that can contain dynamic parameters, pass data between requests, etc.

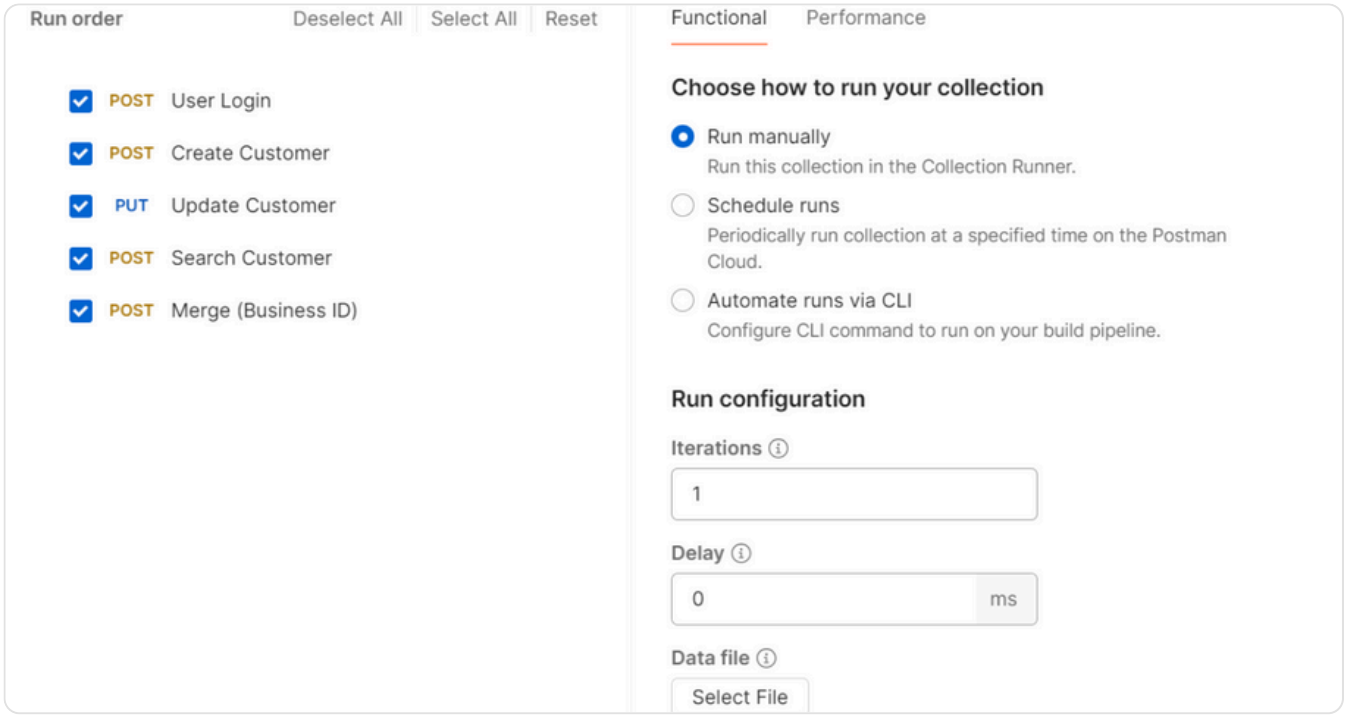
For validation of API, on receiving a response, Postman validates the response as described in the test scripts under “Scripts” section.

Core Features of Postman Automation

1. Running API Collections

Postman allows you to group API requests into collections, which can be run sequentially.

- **Collection Runner**: The Collection Runner in Postman helps you run multiple requests in one go. You can add environment variables, set iterations, and configure delays between requests.



2. Automated Test Scripts

Postman allows you to write test scripts using JavaScript. These scripts validate the API responses by asserting conditions such as HTTP status, headers, and response time.

Example Test Script:

```
pm.test("Status is 200", function () {
    pm.response.to.have.status(200);
});
pm.test("Response body contains sessionId", function () {
    pm.expect(pm.response.json().userInfo.sessionId).to.exist;
});
```

- These scripts run automatically after each request in the collection.

3. Environment Variables

You can create environment variables to manage dynamic data like API keys, tokens, and URLs.

Use {{variable_name}} to refer to these variables in requests.

Example:

- {{baseUrl}} (base URL of your API)
- {{apiToken}} (authorization token)

4. Data-Driven Testing

With data-driven testing, you can run the same test multiple times with different inputs, sourced from CSV or JSON files.

How to Add Data File:

- In the Collection Runner, click Select File and choose a CSV or JSON file.
- Add variables (e.g., {{username}}, {{password}}) inside your requests.
- Each row in the CSV or JSON file will be treated as a separate iteration, and Postman will automatically pass values from the file into the requests.

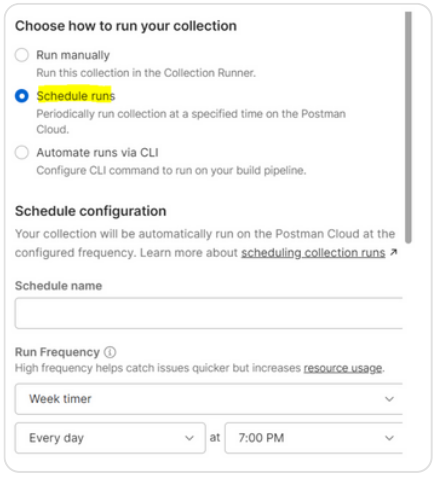
5. Monitors for Scheduling Automated Runs

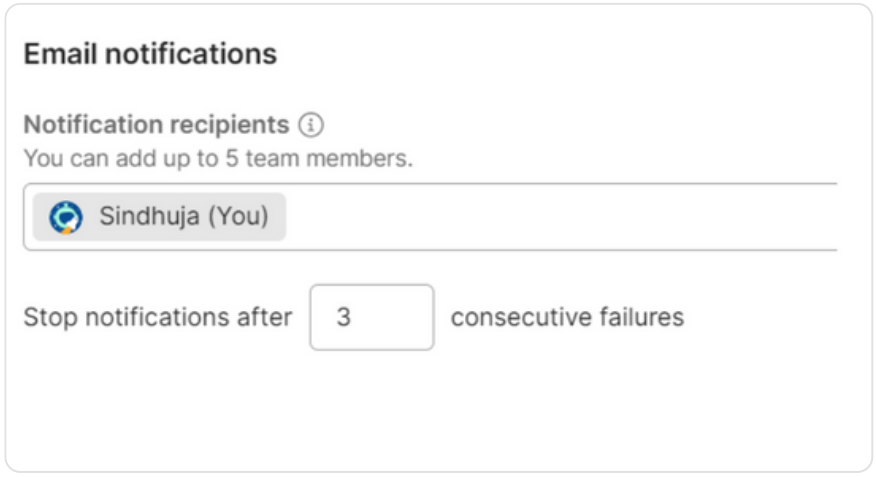
Postman Monitors allow you to run collections periodically to check the health of your APIs.

- You can schedule hourly, daily, or custom runs.
- Monitors can be used for API monitoring, load testing, or checking the status of critical endpoints.

How to Set Up a Monitor:

- In your Run collection, click Schedule configuration.
- Select the frequency (e.g., every 5 minutes).
- Set up notifications (e.g., email reports after each run).





Why we need automation

To run Multiple API services with different data sets in different environments, it will be time consuming to run the API services each time with new data.

The main objective of QA automation is to reduce the combined amount of effort required for manually re-testing of a product which is fairly high.

Also, for removal of the manual testing efforts that are invested in testing a set of functionalities repeatedly.

Set up automated tests to run after any code change, ensuring that new features don't break existing functionality and minimizing the need for manual regression testing, which can significantly reduce the chance of human error by automatically performing repetitive tasks and verifying expected results across various input scenarios.

Here are the key reasons why it's essential:

Faster Testing & Efficiency

- Instead of manually sending requests, automation allows batch execution of API tests.
- Saves time & effort, especially for large test cases.

Consistency & Accuracy

- Manual testing can lead to human errors (e.g., missing parameters, incorrect assertions).
- Automated tests ensure that every request is tested the same way every time.

Regression Testing

- Every time new code is added, you must ensure old features still work.
- Postman automation helps catch bugs early by re-running test collections.

Continuous Integration (CI/CD) Support

- Postman automation can be integrated with Jenkins, GitHub Actions, or CI/CD pipelines.
- Ensures APIs are always tested before deployment.

Data-Driven Testing (Dynamic Inputs)

- Automate testing with CSV/JSON files to send multiple inputs.
- No need to manually enter data for each request.

Load & Performance Testing

- Simulate multiple users calling the API at the same time.
- Helps identify performance bottlenecks.

Security & Error Handling

- Automate tests for authentication, authorization, and error scenarios.
- Ensures APIs handle incorrect inputs gracefully.

How to automate API

To speed up the API testing and to improve the quality, we have automated this testing through POSTMAN.

The three simple automation Steps are given as follows:

- Create all requests with dynamic parameters into a single collection.
- Create test data file which contains input to the dynamic parameters in the request.
- Create test scripts for what we need to validate in the response.
- The test data file can be imported to postman during collection runner execution. The data can be used for validating all the services in the collection.

Postman

Download postman: https://www.postman.com/downloads/

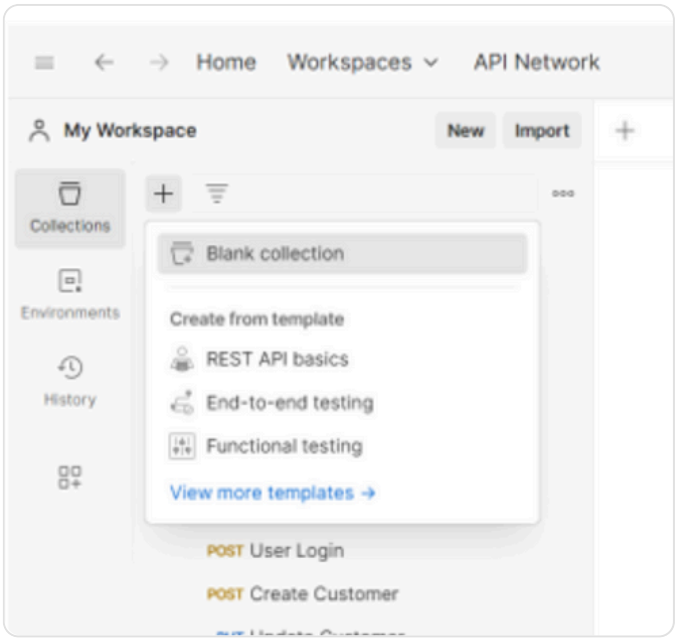
Create an account with an email and sign in.

Let's explain the whole process from creating a collection and running that collection step by step

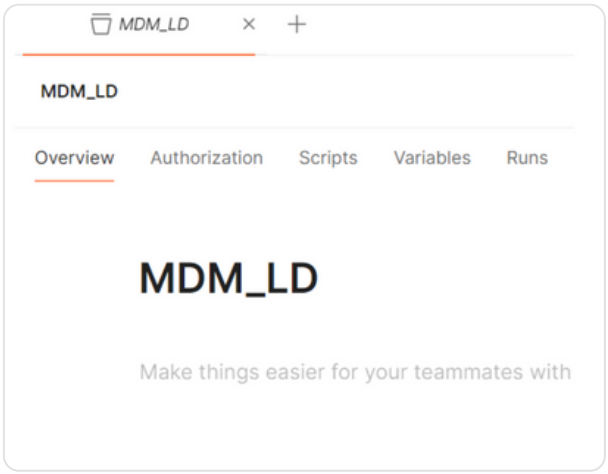
Create Collection

Collection is a set of requests that can be organized into folders. Collections play an important role in organizing test suites. It can be imported, exported and making it easy to share collections amongst the team.

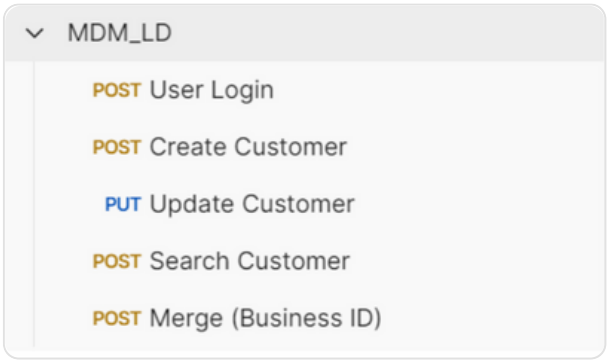
- Open POSTMASTER
- For creating new collection we need to click on the New Collection button and then enter your collection name as well as the complete description for the collection like which type of requests they contain.



- After entering the Description for the collection, enter the create button to create the collection. Now you have your own collection.

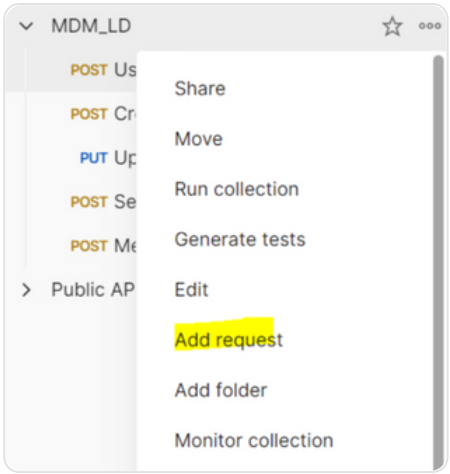


- Create all the requests into a single collection as shown as below.

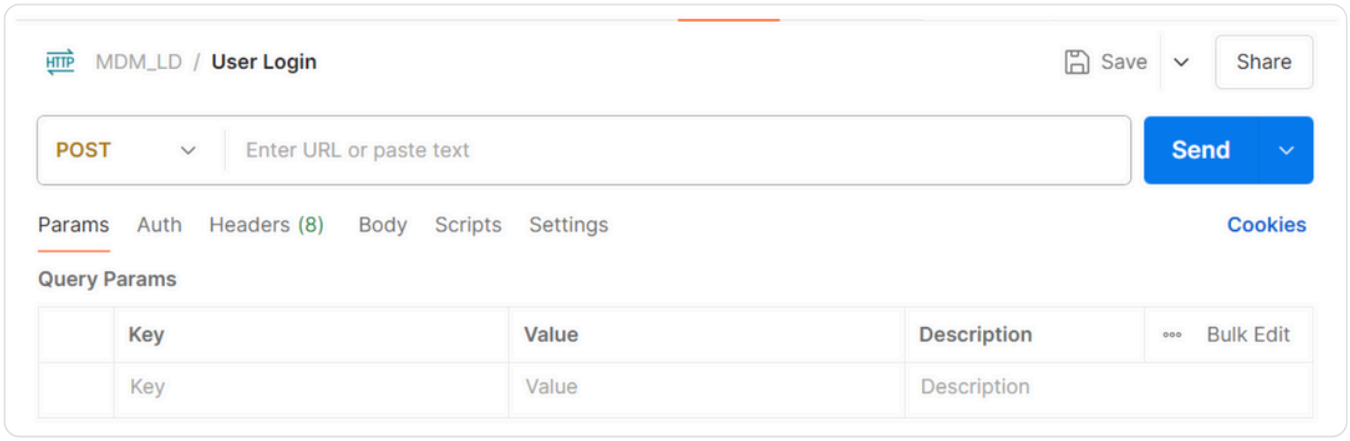


Request Creation

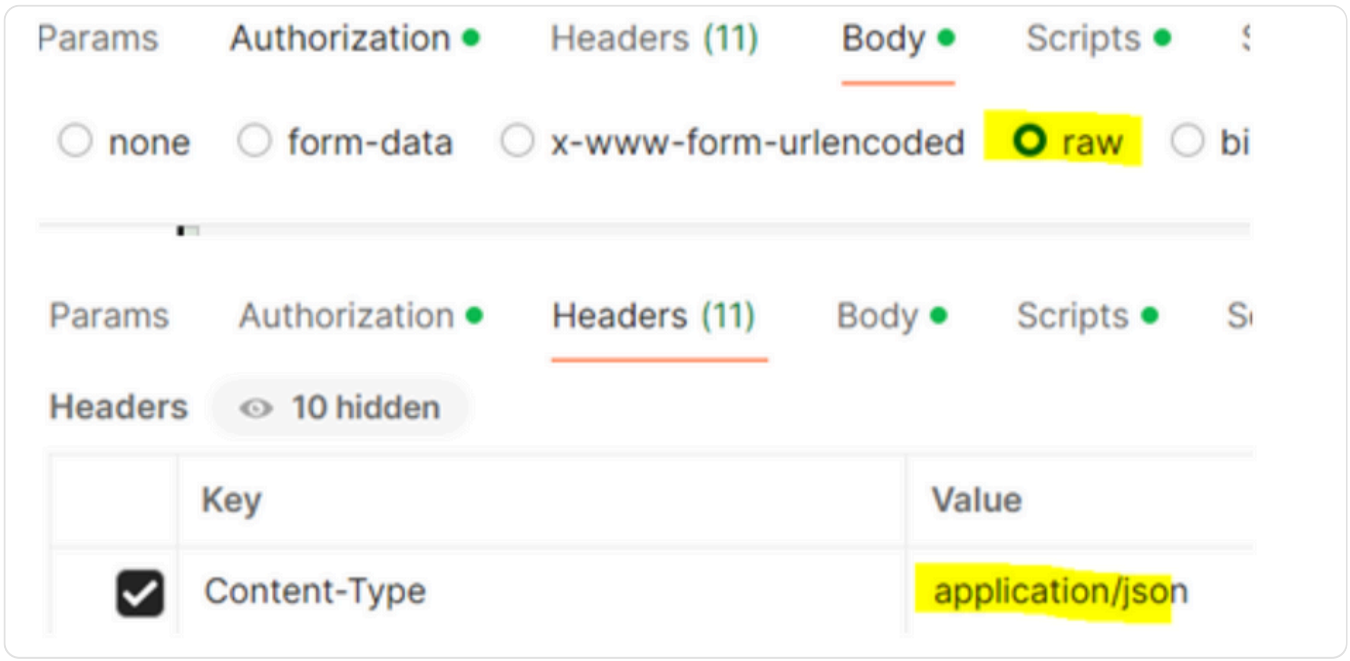
- Select the collection
- To enter the requests into the collection you need to click into the Three dots available along with the collection.



- Click on to the Add Request option to add Request into your collection. Add Request Name, Request Description, and click on the Save button to save your Request.

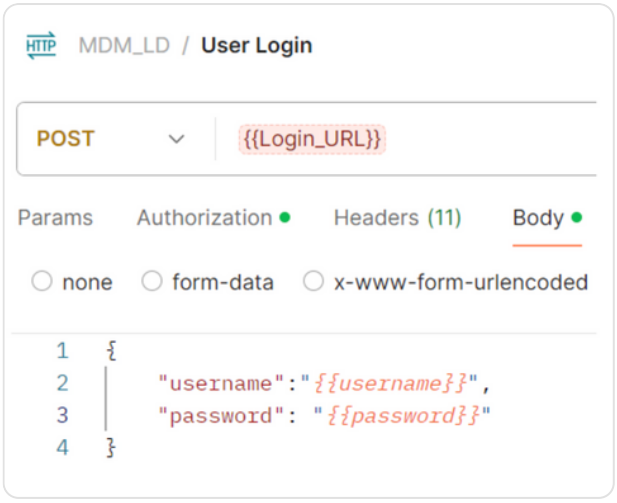


- Click on the added API request and it will show on the Request Panel. Enter the Request URL into that, change the method name according to the API type, change your HTTP request type to POST method
- Go to Body tab and select type as Raw and Select content type for the API as application/Json



- Create your request with parameters and click on the Save along with Send button.

Sample Request: User login request – session id creation.

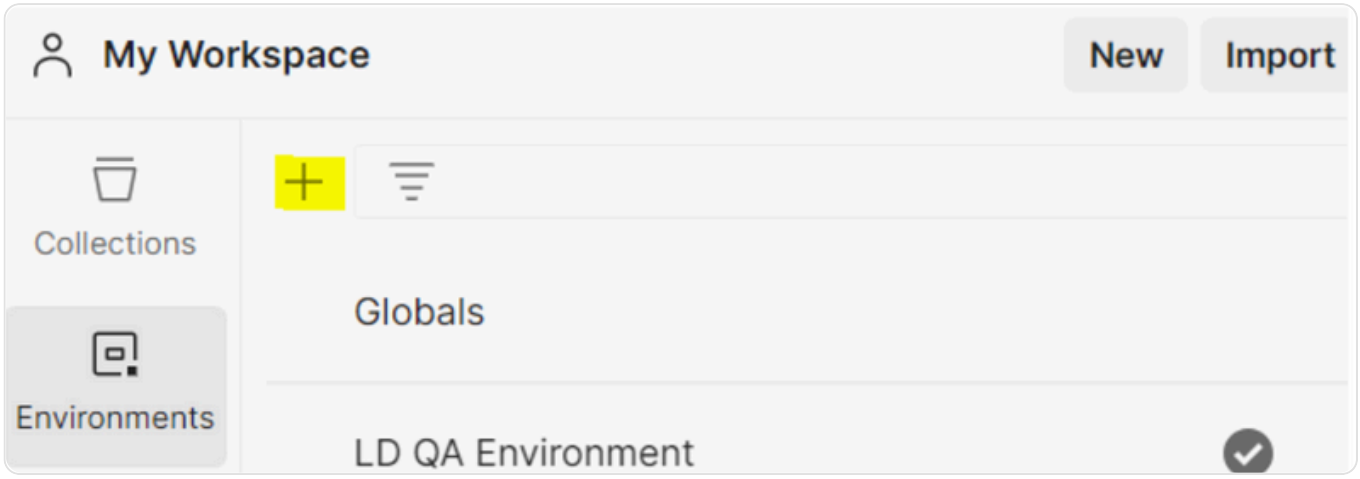


- -- Username and password variables will be provided as environmental variables and Login_URL will be provided in csv file
- Click on Save with your request name.
- So, Add likewise your other requests into the collection.

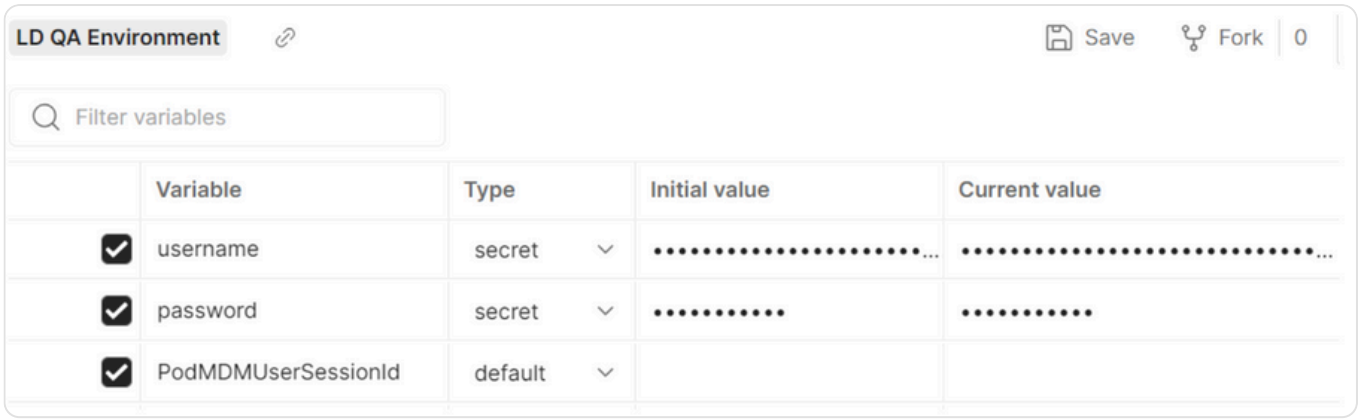
Create Environment variables

There are two types of variables – **global and environment**. Global variables are for all collections whereas the environment variables are defined for a specific set of collections as per the environment which can be selected from a drop-down or no environment can be selected.

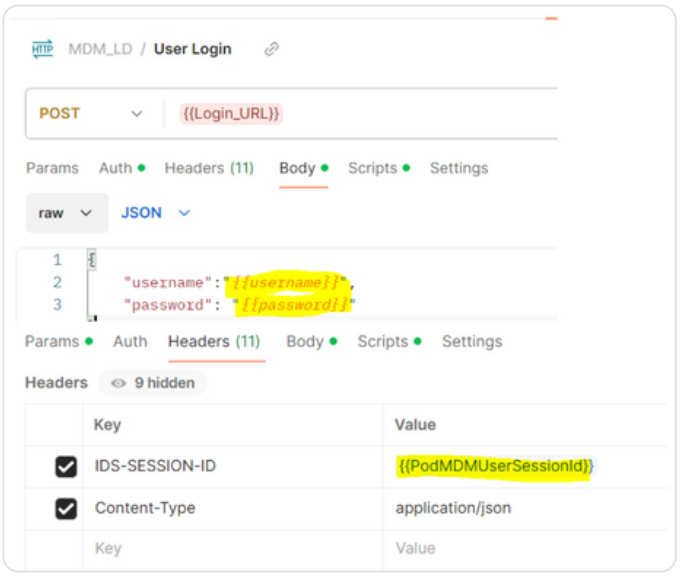
- Here we are creating environment variables
- To create an environment, click on Add button and enter a new Environment.



- For setting a new variable, we need to define the Variable name, Type and Value. Type can be default or secret (to keep its values masked on the screen)



- PodMDMUserSessionId – session id generated from user login response.
- Once defined, variables can be used in request with format surrounded by curly brackets: {{VARIABLE_NAME}} as below



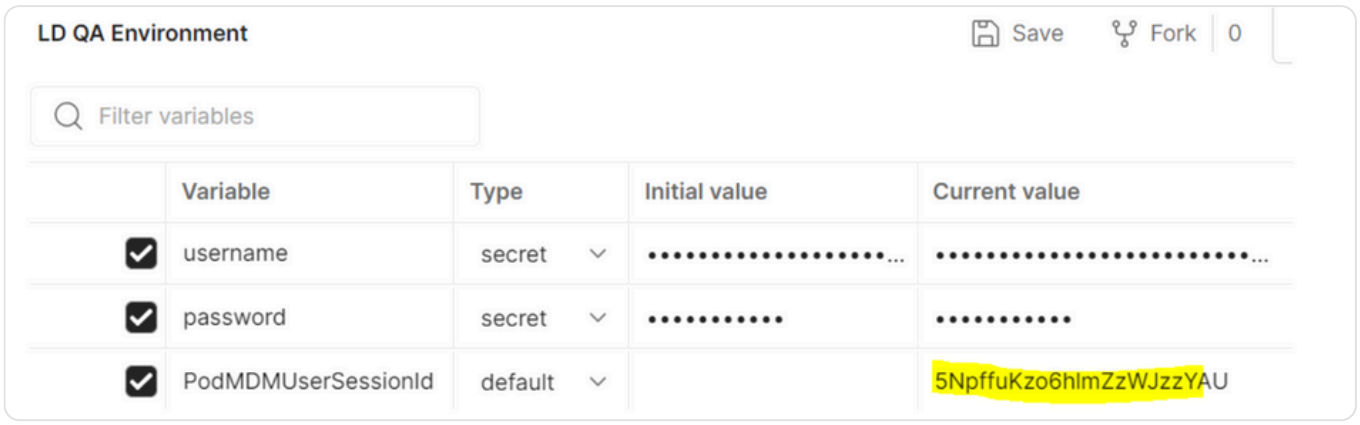
Create Postman Scripts

Tests are scripts written in JavaScript that are executed after a response is received. Tests can be run as part of a single request or run with a collection of requests. In the Postman app, the request builder at the top contains the scripts tab where you write your tests.

- Here we can set Global and Environment Variable dynamically as well.
- Test scripts have been written in each request to verify the responses.

Sample test script:

- pm.response.json() is a Postman script command that parses the response body as JSON.
- jsonData will store the parsed JSON object.
- Properties of jsonData can be accessed using dot notation (jsonData.key)
- const id = jsonData.userInfo.sessionId; //Id stores sessionId, which is inside userInfo.
- pm.environment.set("PodMDMUserSessionId", id); //This extracts sessionId from the API response and stores it as an environment variable.
- console.log("Successful login","\n ", "sessionId:",id); //Used to debug and print values to the Postman console.
- pm.test("Successful login"); // Used to write tests and validate API responses.
- Note: PodMDMUserSessionId – After the request run, observe that keys have been set dynamically by the script as environmental variable and will be used in other requests headers.



The test result will be success when the response meets the expectation which is written in the test scripts else it will be shown as failed.

Create data file

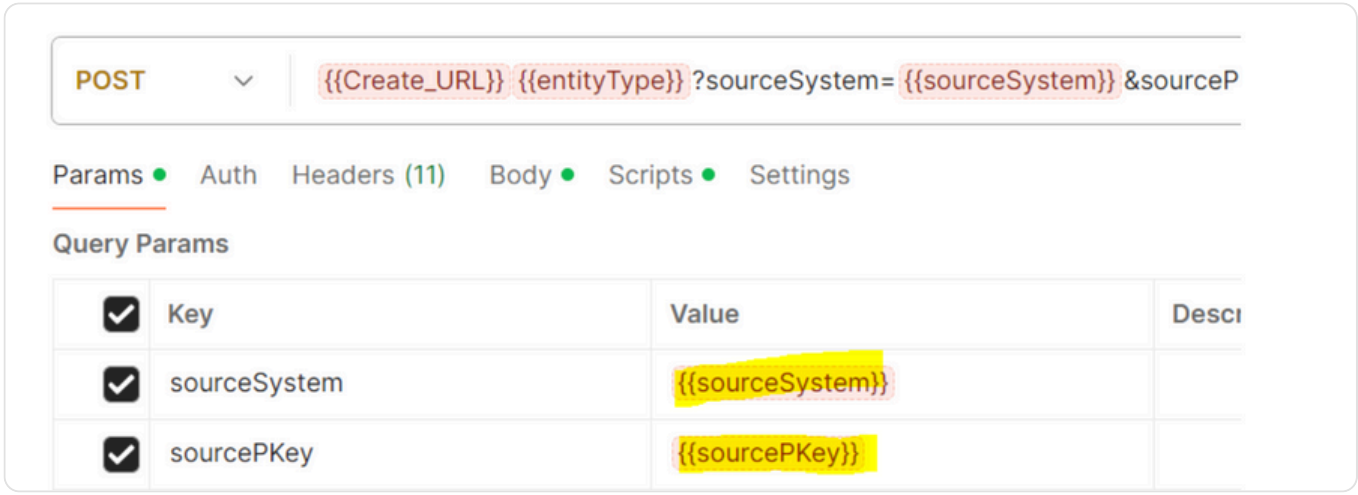
Predetermined value (Hard coded) is never a good practice and will be a pain when the number of test cases are increasing day by day.

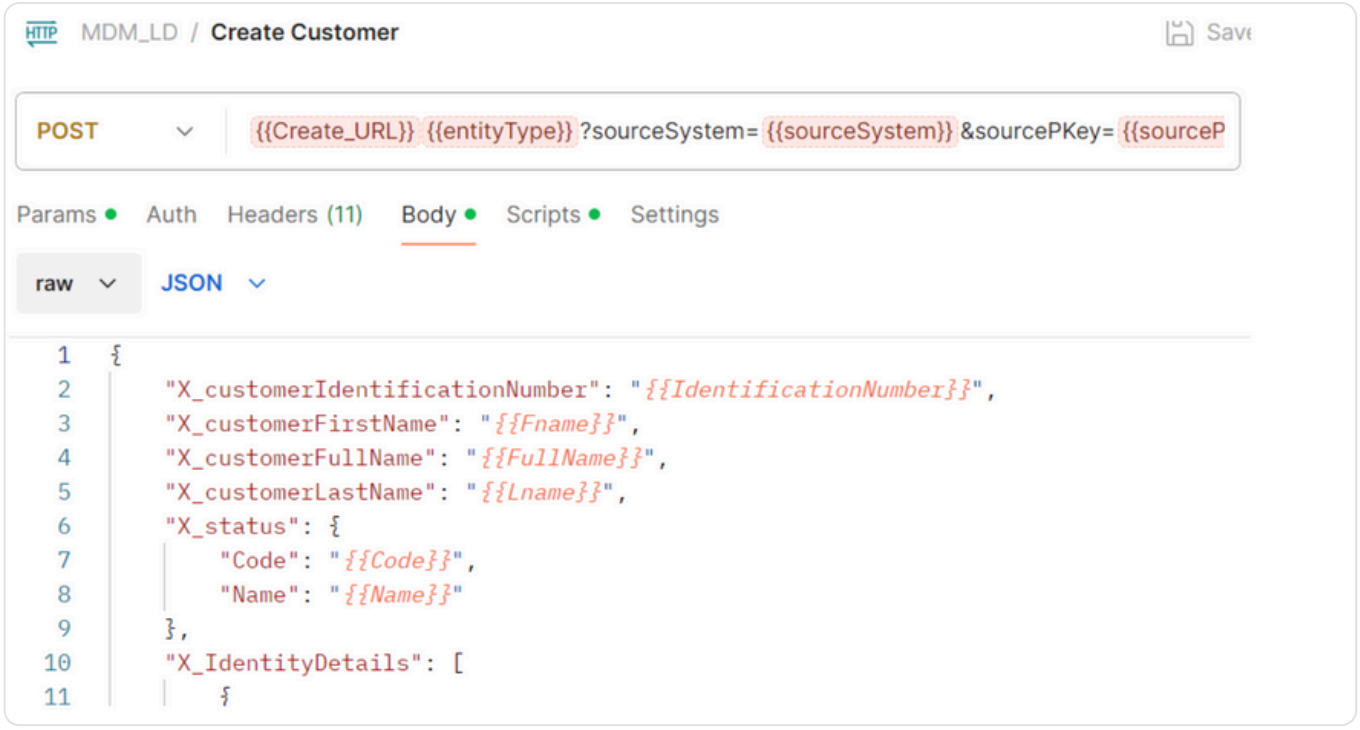
Create a CSV file by providing values for all the variables mentioned in the request. The first row represents all variable names, and subsequent rows represent values for the variables for each iteration. Use the below format to create multiple data for each variable.

Save the CSV data file in any accessible location on your local computer or keep it within your project directory

Login_URL	entityType	Search_URL	customerFirstName	Create_URL	sourceSystem	sourcePKey	Identification	Fname	Lname	FullName
https://dmp	c360_acco	https://usw1	Daniel	https://usw1-r	c360.default.sy	100000	1	Daniel	Sam	Daniel Sam
https://dmp	c360_acco	https://usw1	Williams	https://usw1-r	c360.default.sy	100001	2	Williams	Murphy	Williams Murphy

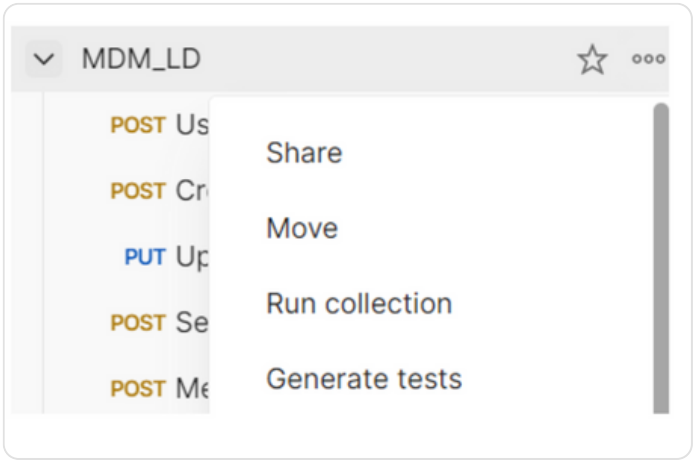
Data can be used in request with format surrounded by curly brackets: {{VARIABLE_NAME}} as parameters and the value in request body as below



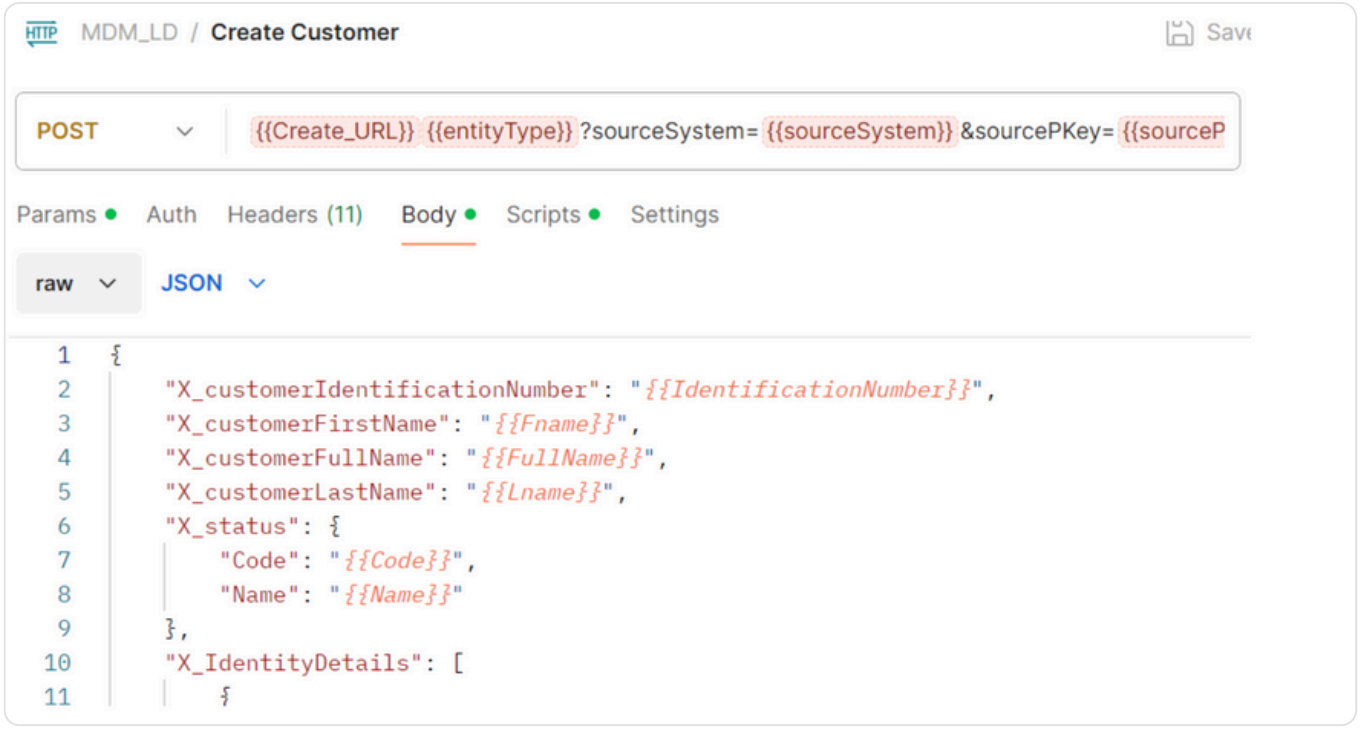


Run a collection with the Collection Runner

The Collection Runner allows you to run multiple API requests from a collection in sequence, automating testing and workflows. To run a collection in the Postman, click on 3 dots next to the collection's name and select Run Collection.

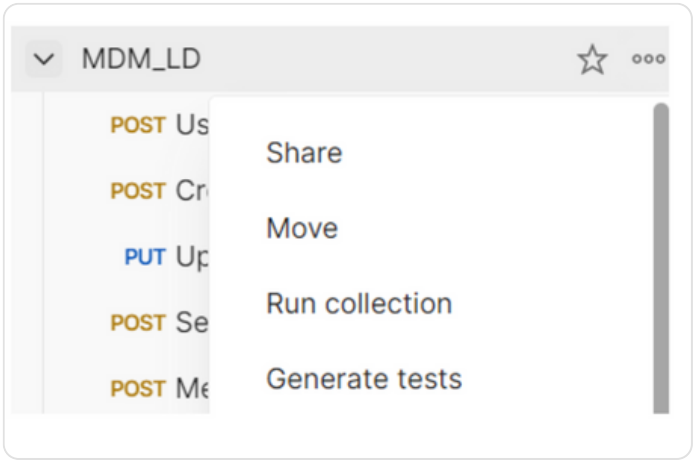


Runner page should appear as below.

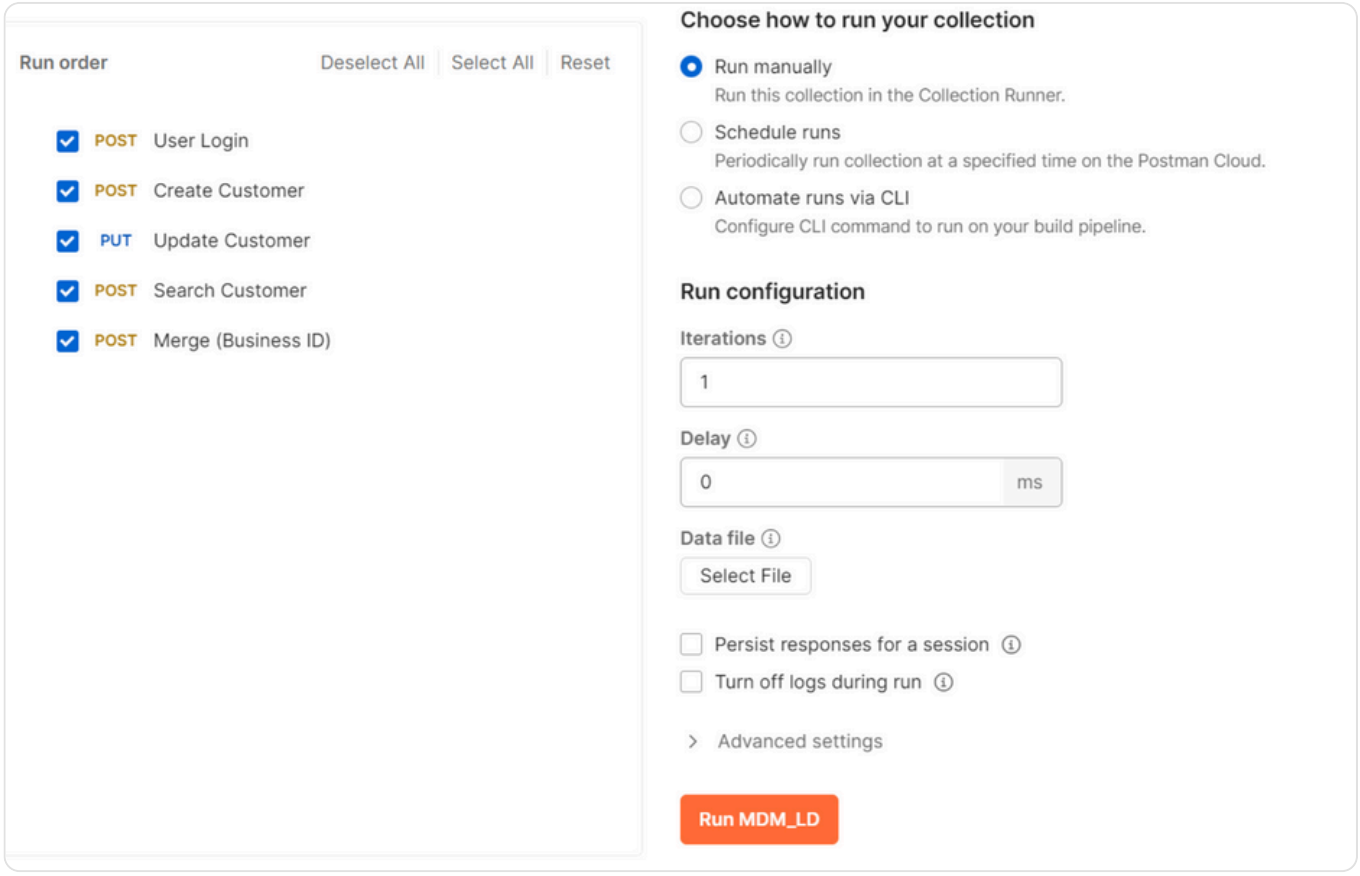


Run a collection with the Collection Runner

The Collection Runner allows you to run multiple API requests from a collection in sequence, automating testing and workflows. To run a collection in the Postman, click on 3 dots next to the collection's name and select Run Collection.

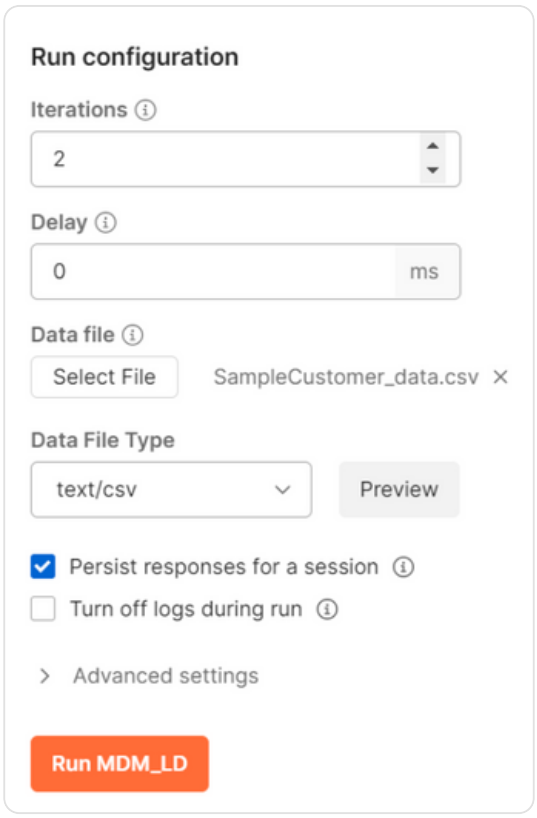


Runner page should appear as below.



Run the Collection by setting up the following:

- Select the required requests to run as a collection
- Choose how to run the collection – Manual or schedule runs
- Select Run manually to run the collection in the collection Runner
- Select Schedule runs to run the collection at a specific time on the postman cloud
- You can run the collection multiple times by setting the number of iterations. In this sample, iteration has been set as 2 as per the data present in input file
- To avoid rate limits or API blocking, add a delay between requests. If needed we can set delay time. Eg. Set delay as 2500 ms
- Click "Select File" and upload your CSV file
- Click on Run MDM_LD button

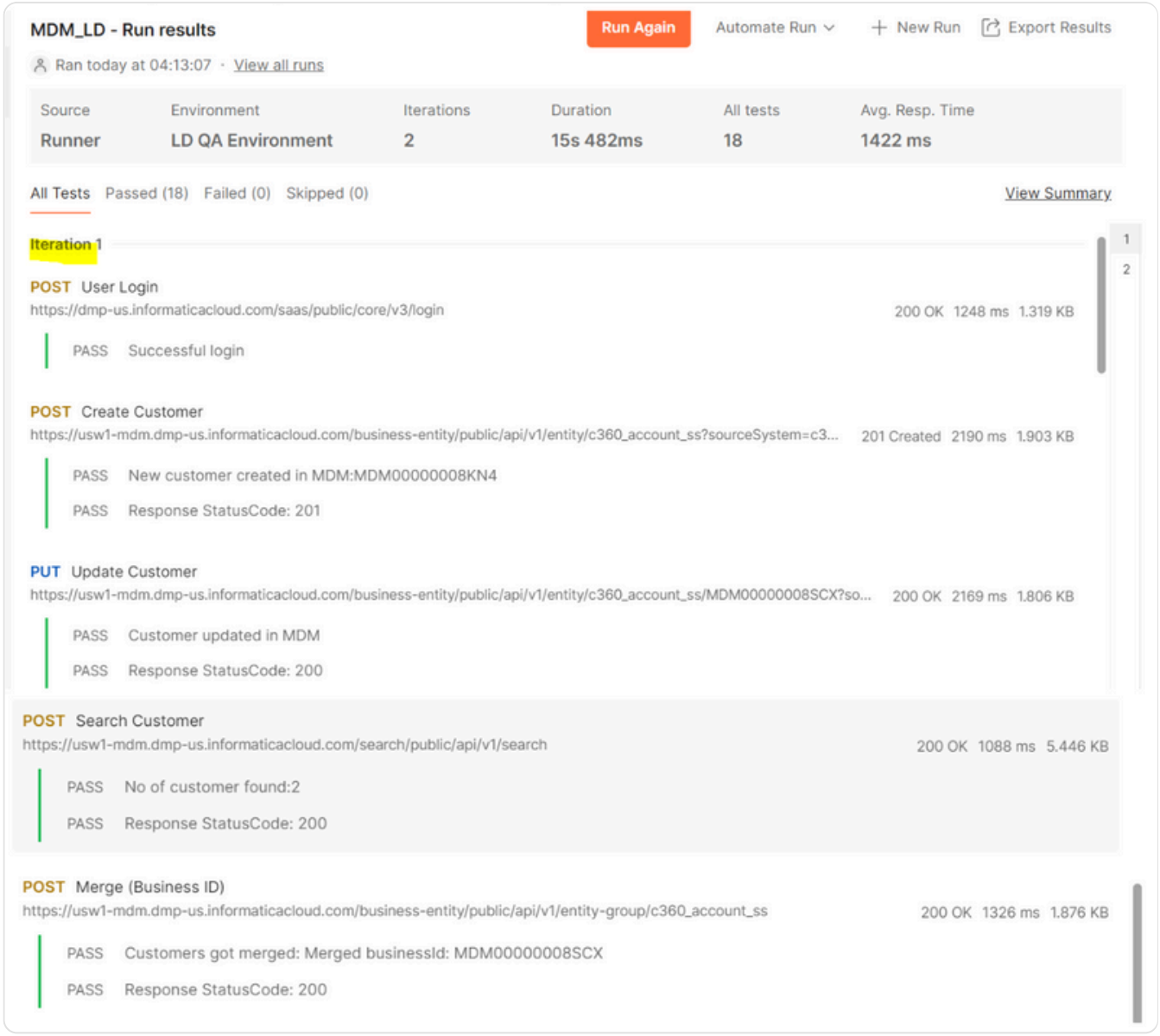


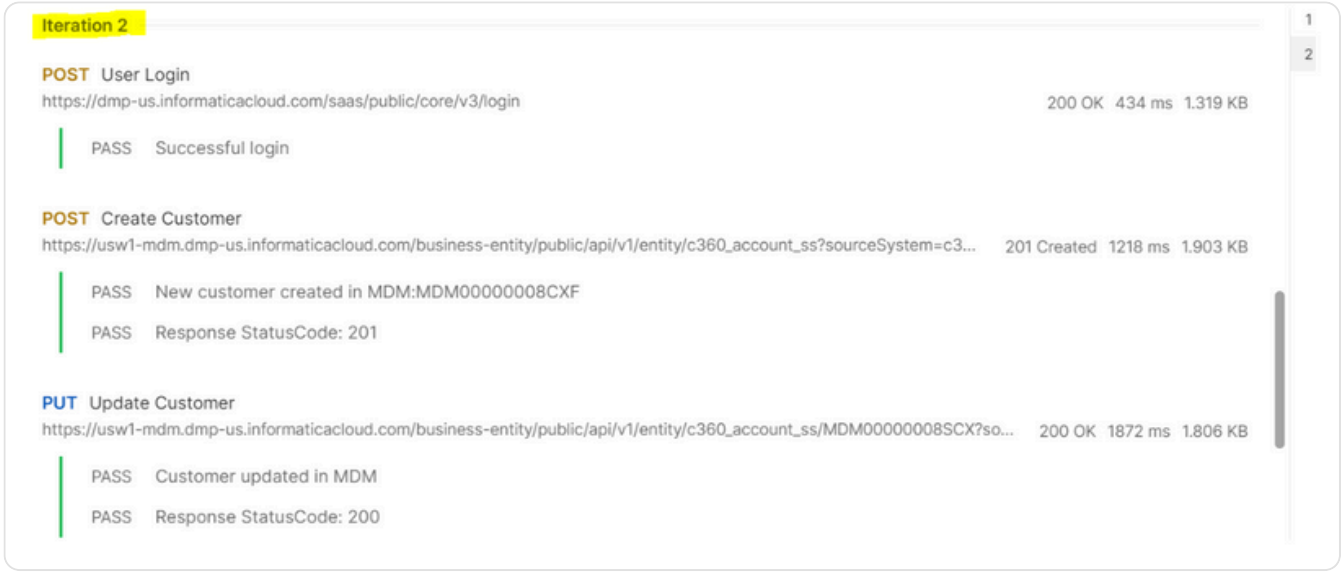
Run a collection with the Collection Runner

When you run a collection, the collection runner displays the results for all tests. The test results include the response time in milliseconds and details about whether a specific request in the collection passed or failed its tests

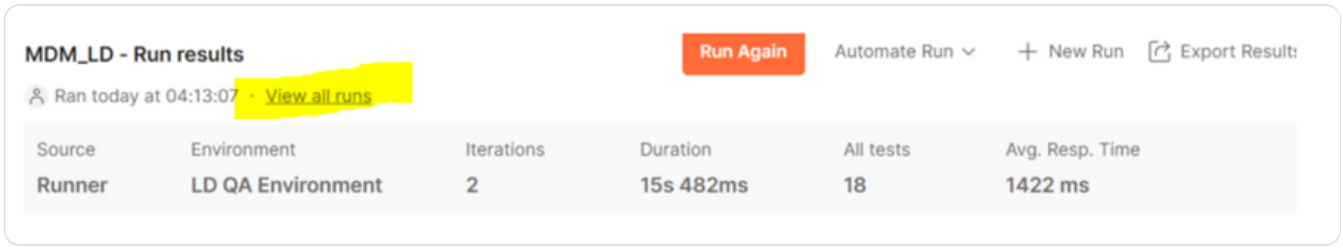
- Run Results page should be displayed after clicking the Run button. Depending on the delay, you should see the tests as they execute.
- Once tests have finished, you can see the test status if it is passed or failed and the results per iteration.
- We can export the test results by clicking Export Results button.

The Run results displays the iterations, response time in milliseconds and details about whether a specific request in the collection passed or failed its tests

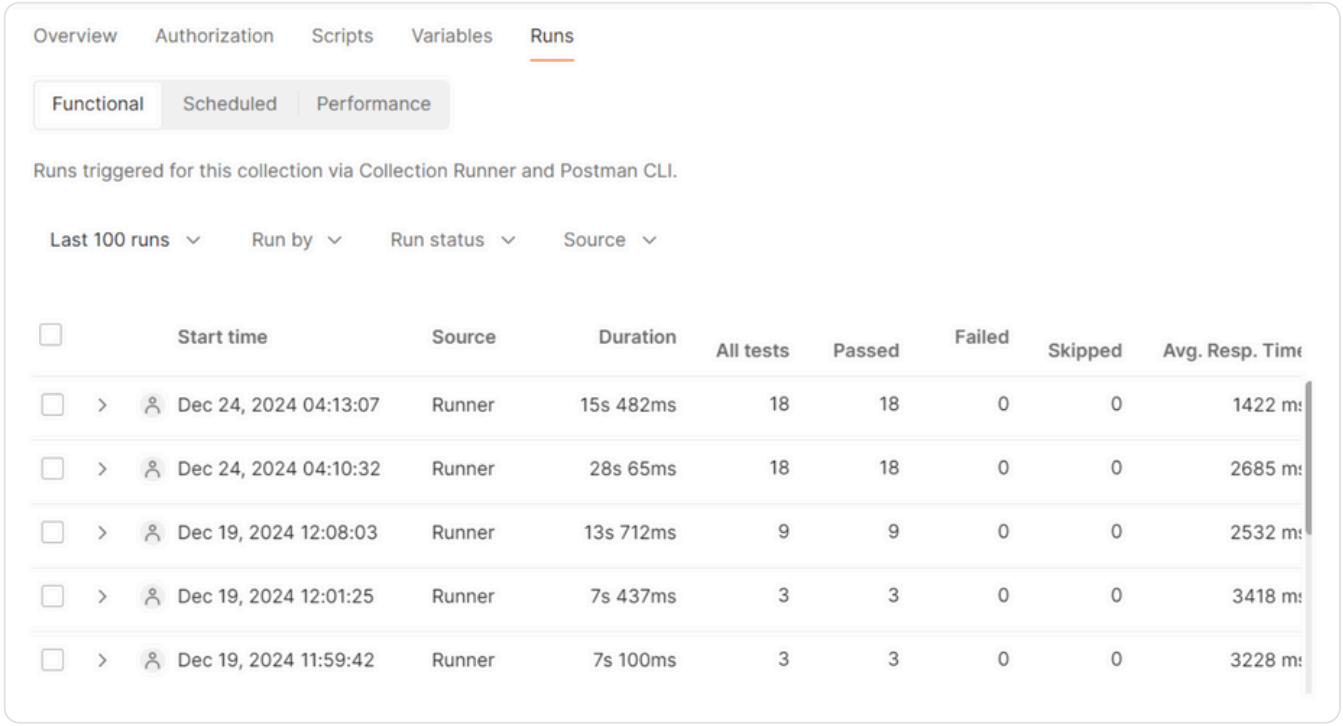




Click view all Runs to see the run history



The below screen displays all the runs triggered for the particular collection with duration, pass/ fail status.



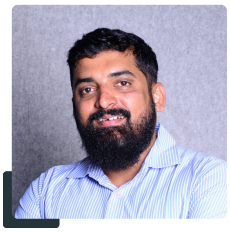
The same steps can be followed for any number of requests and environments and data set.

Sample Postman collection and sample data:

MDM_LD.postman_collection.json

SampleCustomer_data.csv

Authors



Jayachandra B.
Technical Lead QA - Level 2

Sindhuja Palaniappan
Technical Lead QA - Level 1

About LumenData

LumenData is a leading provider of **Enterprise Data Management, Cloud & Analytics** solutions. We help businesses navigate their data visualization and analytics anxieties and enable them to accelerate their innovation journeys.

Founded in 2008, with locations in multiple countries, LumenData is privileged to serve over 100 leading companies. LumenData is **SOC2 certified** and has instituted extensive controls to protect client data, including adherence to GDPR and CCPA regulations.

LUMENDATA

Get in touch with us:
info@lumendata.com

Let us know what you need:
lumendata.com/contact-us