MEC EVERYAPI TEST ENGINE – USER MANUAL

Contact:

sales@testonneed.com

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
0.1	Sairam C.	First Draft version	

Review & Approval

Requirements Document Approval History

Approving Party	Version Approved	Signature	Date

Requirements Document Review History

Reviewer	Version Reviewed	Signature	Date

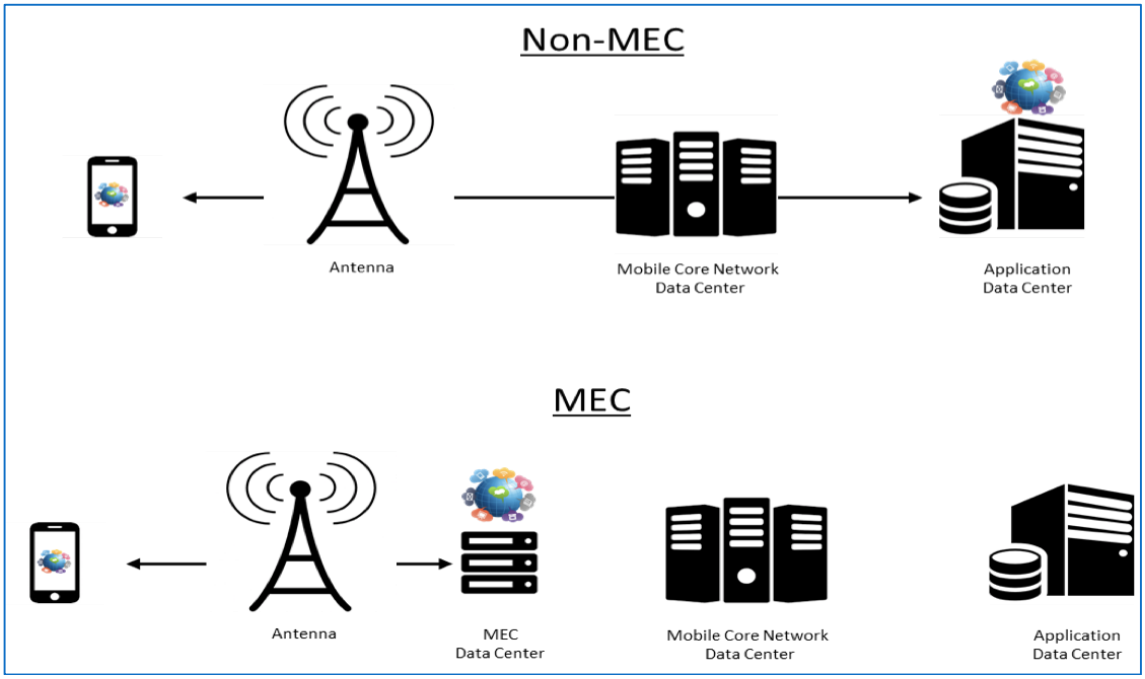
Table of Contents

1. INTRODUCTION

Multi-access Edge Computing (MEC), also described as a Mobile Edge Computing is an industry innovation that improves the performance and intelligence of the software application that deliver services such as IoT (Internet-of-Things) to end customers.

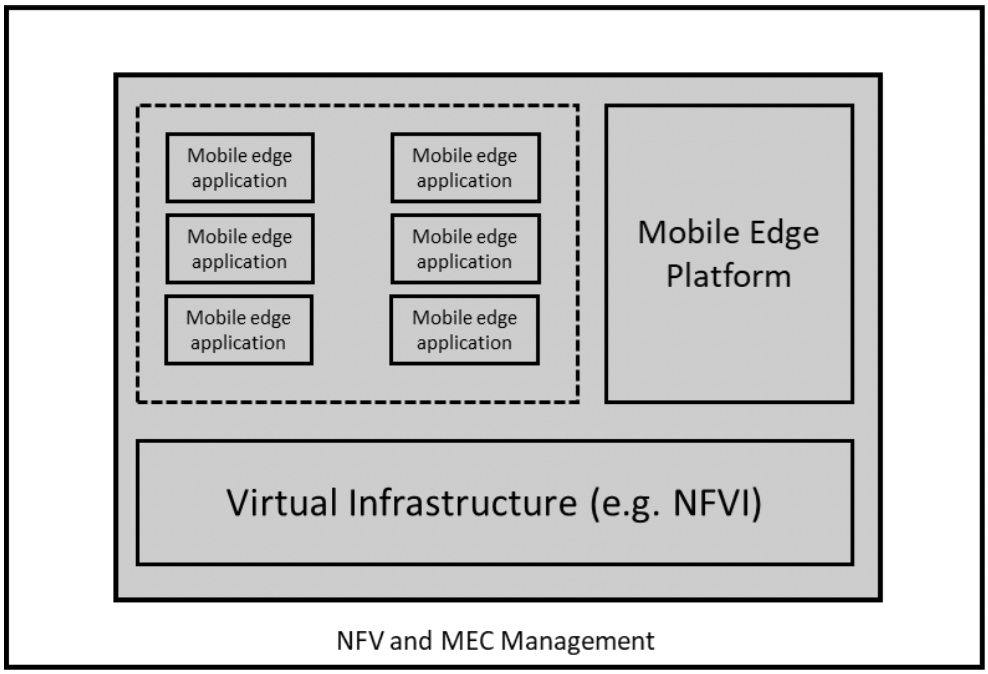
The improved ultra-low latency and high bandwidth performance are made possible by the deploying applications in the data center at the edge of the network closer to the customer. The improved intelligence by real-time access to radio network information that can be leveraged by applications.

The picture below shows the architecture of the systems in action. The sole intention of the MEC architecture is to provide the best experience for the user by removing the latency as much as possible thus getting the applications closer to the user so the user does not need to connect to the server which will take a lot of time.



2. ARCHITECTURE

The MEC technology is supported by the edge data centers and the deployments are achieved as defined by European Telecommunications Standards Institute (ETSI) in standard specifications ETSI GS MEC 003 V1.1.1 (2016-03). The implementation requires the MEC platform that hosts and manages MEC applications on Network Functions Virtualization (NFV) infrastructure. The MEC and NFV management manages its respective domains as shown in the picture below.



The next generation technology such as MEC and NFV requires protocols to communicate between antenna (eNodeb), MEC platform, MEC applications, NFVI and management entities. The connectivity among these entities is called “Reference Points”.

ETSI defines the API, data model and data format in the specification for these reference points.

3. WHAT WE DO

"What if testers have a solution that provides lot of test cases to ready- to-go client and server side API conformance test cases?"

TestOnNeed offers precisely such a solution to our customers who are ready to test at client side and server side.

We provide a ready to use solution to the customers which help them to achieve quick results for testing the client side. This solution will help to detect problems at an early stage which will reduce the cost of the product development cycle.

We provide:

1. MEC EveryAPI Conformance Test Suite : In this document we will be describing the Conformance solution in details. This conformance test suite solution will help the users understand and validate the request and the response for the MEC applications.

2. MEC EveryAPI Test Engine – This solution is concentrated on the creation of new APIs using the engine which can be used by the customer to create the Client and Server code for any MEC specification using the YAML file of the particular specification. This then validates the queries sent from the client to the database and checks the expected response to the actual response.

4. PREREQUISITES

The installation of the application and setup which have been used to achieve the automation solution, please refer to the installation videos below.

https://www.youtube.com/watch?v=gJlOaAMAPWU (Windows)

https://www.youtube.com/watch?v=PXRJ4VG4HAk (Linux)

Listed below are the required software applications. All the applications being used are open source.

- **Postman**

  This is used to record and automate the process of posting requests to server and fetching the corresponding responses back while the user will be executing the actions.

- **NodeJs**

  This will be used by the solution for the generating the requests which will be used by the solution for execution of test cases.

- **Test Link**

  This will be used by the testing professional to keep a track of the test cases and specification document. The test plan will be provided to the customer and can be imported to the application after installation.

- **Sublime Text**

  This will be used as an editor to view the test data by the test professional.

- **Mongo DB**

  This is the tool used for the NoSQL Database whicg is used by the Test Engine to store data into and fetch data for test case execution.

- **Robo 3T**

  This is used as an interface to the Mongo database server which will help the user view the data stored in the database in tabular and JSON formats.

5. DESCRIPTION OF THE SOLUTION

The deliverable package contains a set of a python compiled files which make up the Test Engine. The Test Engine code accepts MEC YAML files as input to it and provide multiple files as output. The whole engine will create a Postman Collection JSON file which will contain the client code and will be imported to Postman application for execution.

The customer will be provided with the Server end code which will be able to handle the requests sent from the Postman application.
The backend requests will be generated automatically and the API mapping will be performed by parsing the YAML file provided by the user.

Here the user will be able to generate mapping API for any MEC specification as per the current deliverables.

During the execution the requests will be generated from the Postman application and will be executing the test cases one by one as mentioned in the Test Link test case document.

The Postman Application will extract data from the database for each test case execution. The database tables will be provided to the customer as a part of the deliverable package.

The data from the database will be fed to the application for execution of the test cases. The solution will be able to take the requests mentioned in the collection file which will be imported to postman.

The request will be served by the MEC server and will send the response to the client end which will then be validated based on the parameters like data type, cardinality.

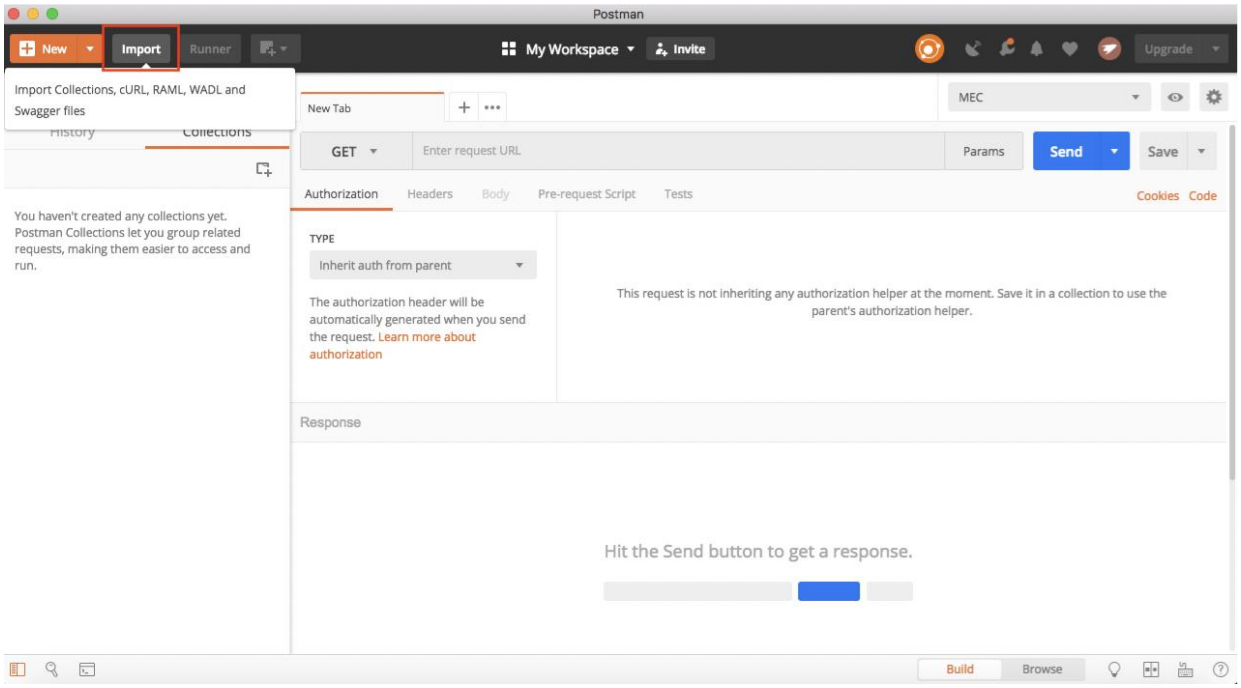
The customer will also get a set of links of videos which they can follow and execute the test cases with ease and reach the market quickly with a quality product.

6. PERFORMING CONFORMANCE TESTING

a. The deliverable package will contain a set of Python code which make up the Test Engine. The test engine is capable to creating the client and server end code for any MEC specification when a YAML file in the supported format will be fed as input to the Test Engine.

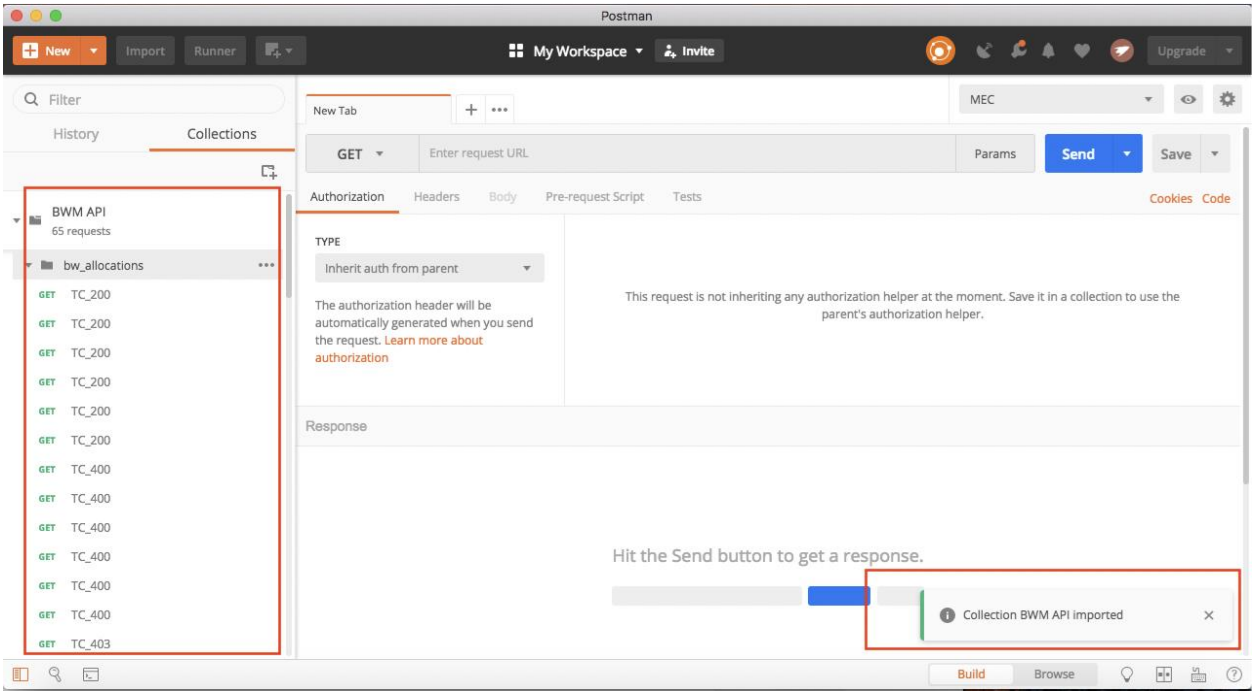
b. The created Postman collection file then can be used by the user without any modifications required to be made to it.

The collection file will then be required to be imported to Postman application.



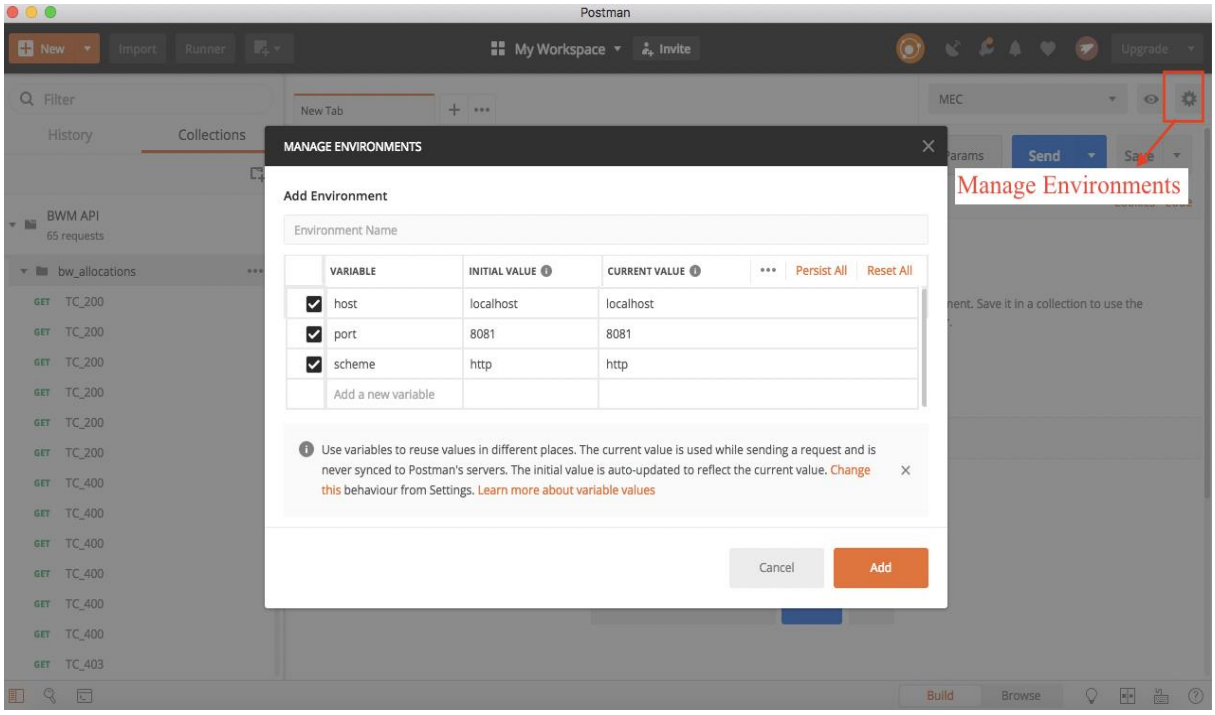
c. After importing the Postman collection JSON file in to the Postman Application then left side section one folder will add automatically and user can see the message as a imported successfully.

On expanding the view folder, the user will be able to view the entire set of test cases which are built based on the input file specifications. The user here will be able to only execute the ready-made test cases and cannot build any test cases from any test specification document.



d. The user will have the option to choose the environment to run the test cases on. On each test cases the user can provide the Environment global variables and the respective values fofr the application environment on the Postman application.

The below image depicts how to set up the Global Param's and values in the Postman Application.



e. After Adding the Global Variables and Values, user have to select the Environment Name in the Dropdown from the right side of Postman Application.

After setup Global Values in the Postman Application the user will be able to start the execution. Postman will use the Mongo database to extract the data to be used for each test case execution. Mongo will be also used to store the data from the test case execution.

f. Postman application will execute the test cases mentioned in serial order. The data for the execution will be taken from the data sheet which will be provided to the user in the deliverable package. The data will be prefilled has a specific order too. The automation solution will be able to extract the data only if the order is not changed.

The user will be able to add to existing set of data as per their requirement and execute the test cases.

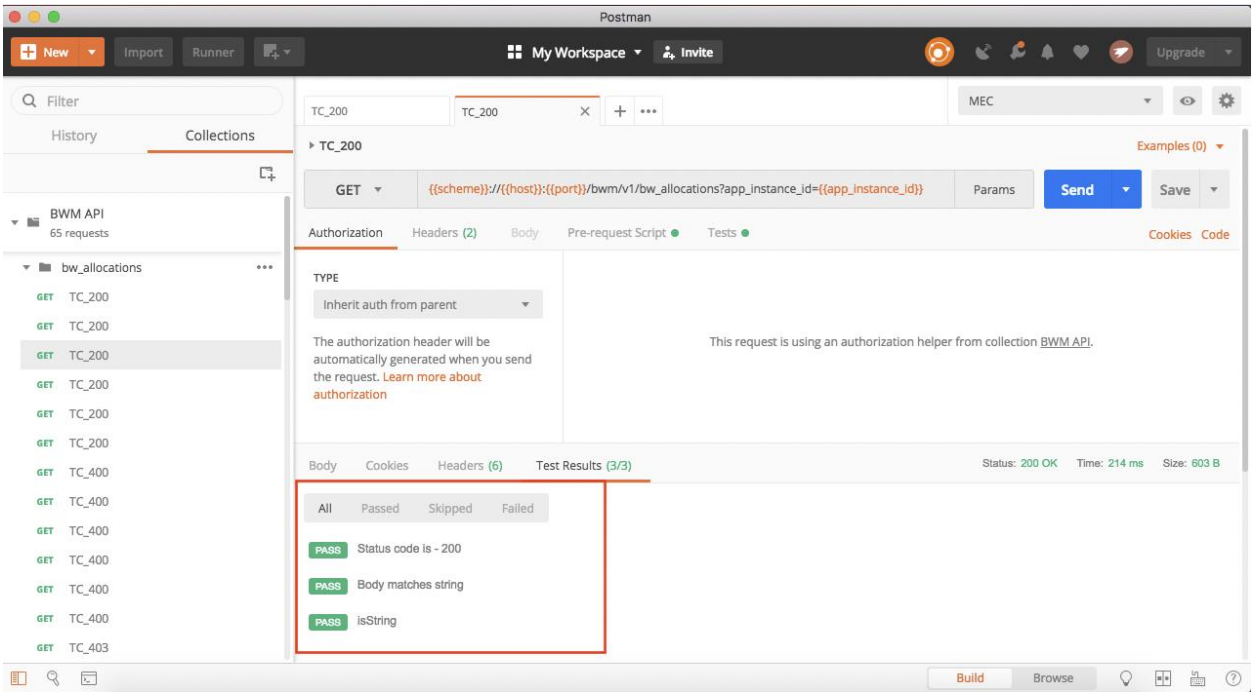
g. The user can execute the test cases in two ways:

1. A single request can be executed by clicking on the “Send” button. The user will be able to view the response in the response section on the Postman application.

2. The user will be able to execute the entire set of test cases by clicking on the “Runner” button and then setting the environment, choosing the collection to execute (in case of multiple collections imported to Postman) and then will be able to view the results in the results section.

h. The user will be able to view the results of the execution as shown below. The execution status will be shown on the Postman application and will also be stored in the Mongo DB for each and every execution.

The picture below shows Test Results in Postman Application.



Please click on the Below YouTube Link for Video: -

https://www.youtube.com/watch?v=3P_LVIdGZNc&feature=youtu.be&list=PLfDSBK5ZoW6SK0wCQnvkZ3hLM9%20ELjeJX5

7. ABOUT US

TestOnNeed is an on-need open source test automation and DevOps solution provider which aims to better the speed, scale, coverage, and quality of software application products. We live only to help businesses seeking the winning formula to bring their products, and services to the market fast to beat the slow. We do this by helping our customers to create flawless, high quality, high-performance Block chain, Artificial Intelligence (AI), Internet of Thing (IoT), Multi-Edge (Mobile) Edge Computing (MEC), 5G, Cloud, Micro services, Augmented Reality (AR), and Virtual Reality (VR) software application products. Moreover, we make it all happen with open sources using an open source testing ecosystem.

At TestOnNeed, we don't just test products; we make products better. Being fast is all about transformation and success is all about welcoming it. To discover more, visit us at https://testonneed.com/

If you need additional information or have questions about our solutions, demo or purchase our solutions, please reach us at mailto: sales@testonned.com.