# rmm-services-server-app

Challenge of Ninja One

# Summary

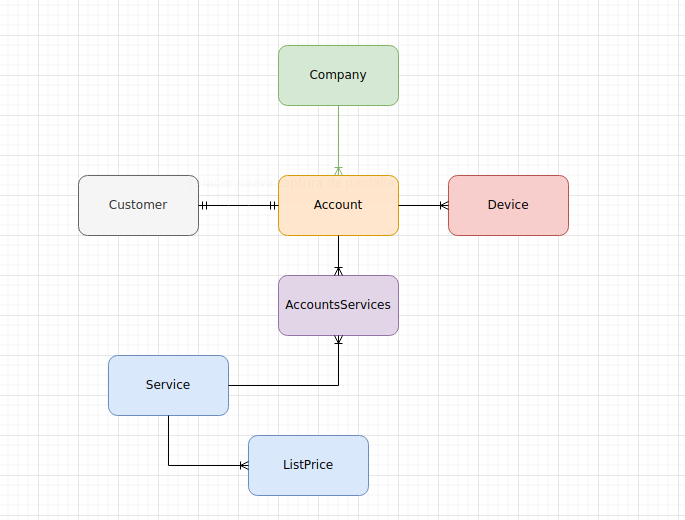
This project is a implementation about the Ninja One challenge.

In this project we need create a Spring Boot Project, with microservices architecture

# Used technology

* Spring FrameWork
* Spring Boot
* Spring Data JPA
* Spring Web
* Spring Security
* JSON Web Token Support For The JVM
* Postgresql
* PostMan
* JAVA

After to read the challenge, I created my ER Diagram as a solution, show below next picture

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/diagramRmm.png)

# Setup and run Project

next explanation is how setup and run this project

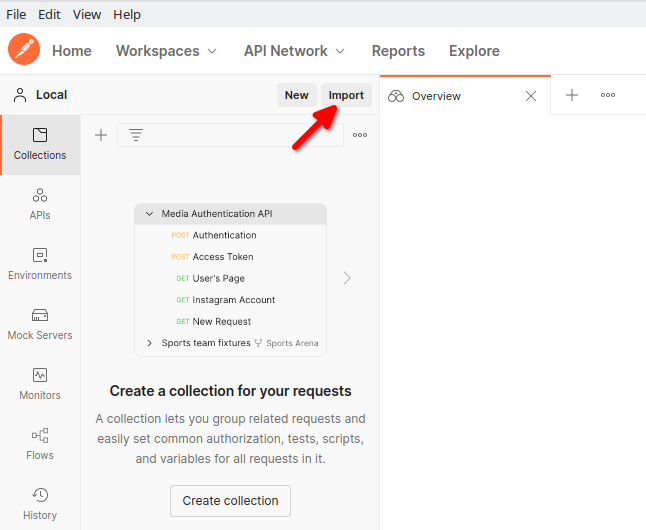
* git clone this repository
* Open project with Intellig IDEA, eclipse, netbeans or visual studio code.
* For run this project should have install and setup Java 11 in your machine, and in your IDE that prefer.
* For run this project should have install postgres, also create a database with name -> "test" and password -> "password".
* last run application, when this run to create all tables.

# Import WorkSpace POSTMAN

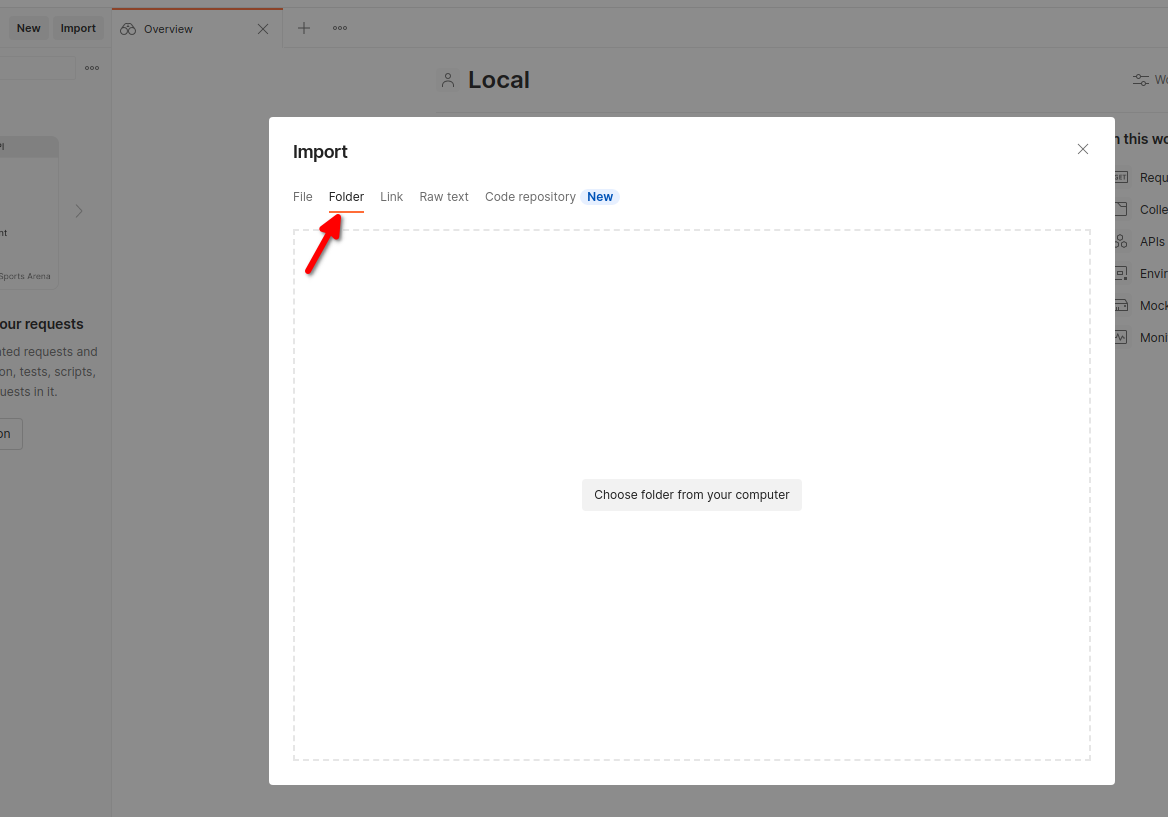
Inside this project in the folder rmm-services-server-app exists and file .zip with name workSpacePostMan.zip, in this workSpace has all request that need to test project.

Import WorkSpace inside PostMan Decompress workSpacePostMan.zip, before From PostMan UI import to folder as show next images

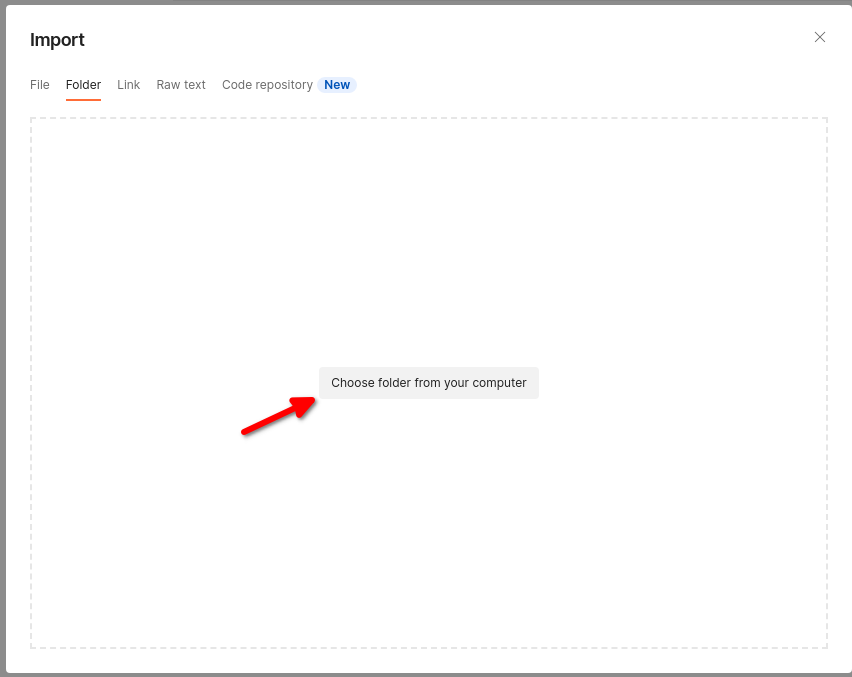
Pulse "Import" button

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/pulseImport.png)

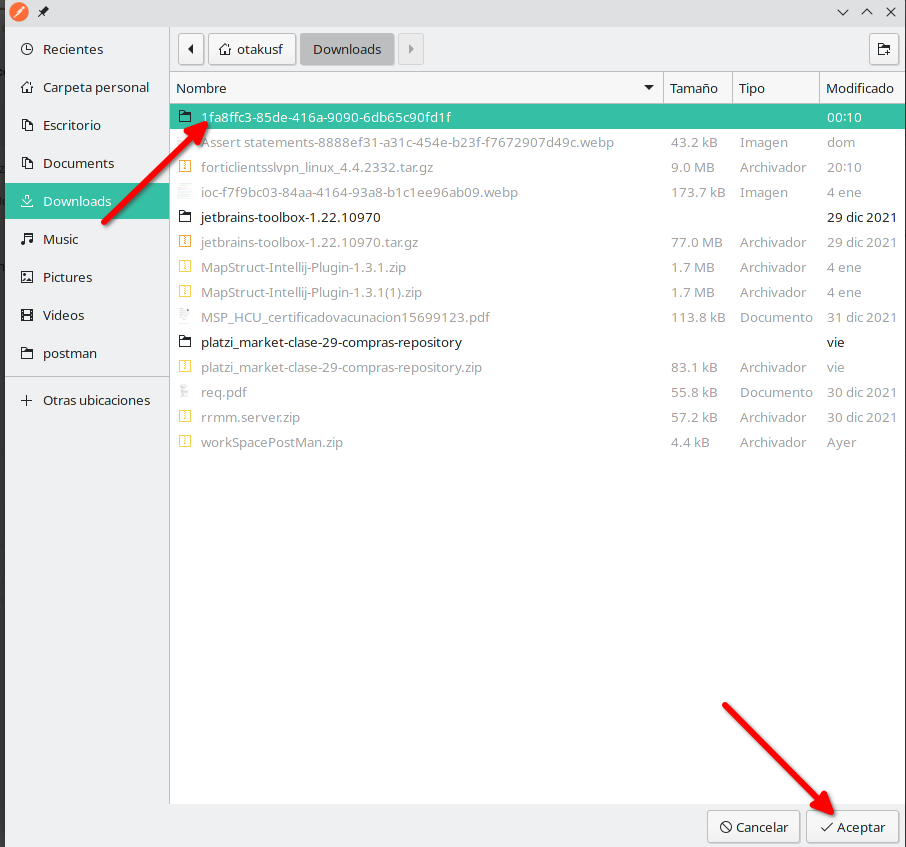
* Pulse "Folder" Tab

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/pulseTabFolder.png)

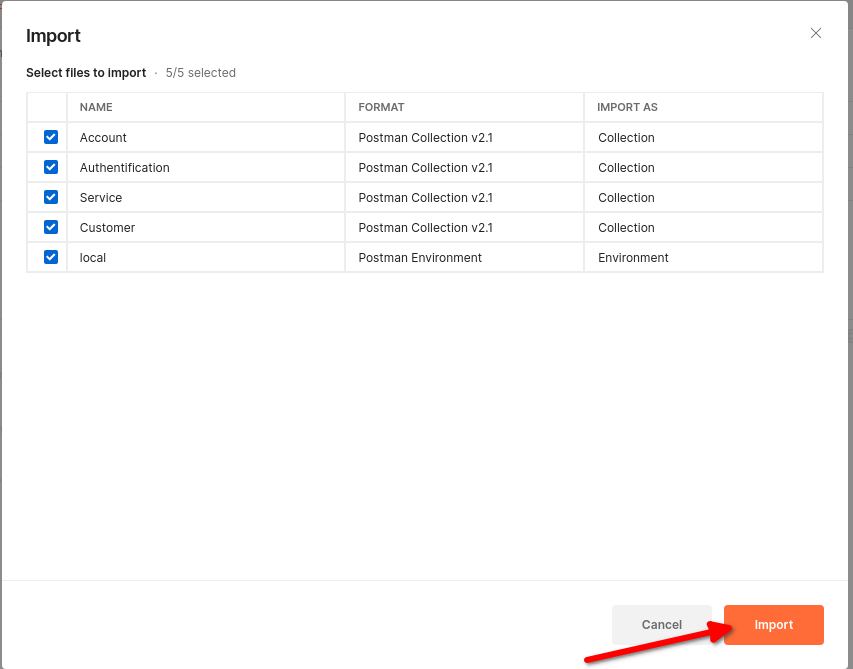
* Pulse "Choose folder from your computer" button

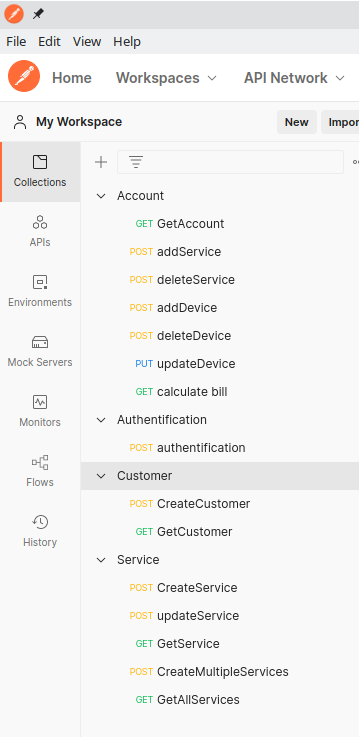
[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/pulseChoose.png)

* After you choose decompress folder.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/selectFolder.png)

* Select Collection

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/selectCollections.png)

* Show all request import
* [](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/importColletion.png)

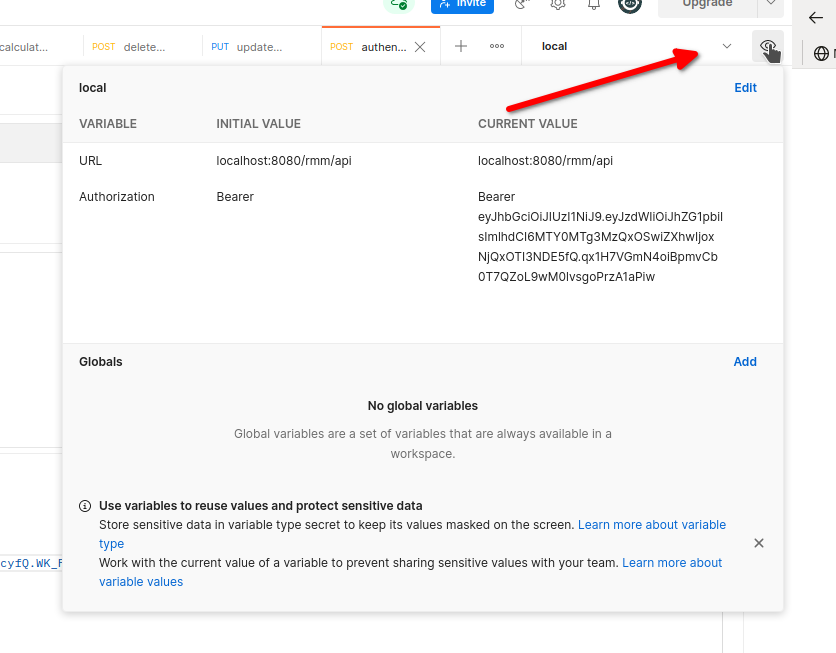
Once the workspace is loaded, you can setter test data

# Upload data test

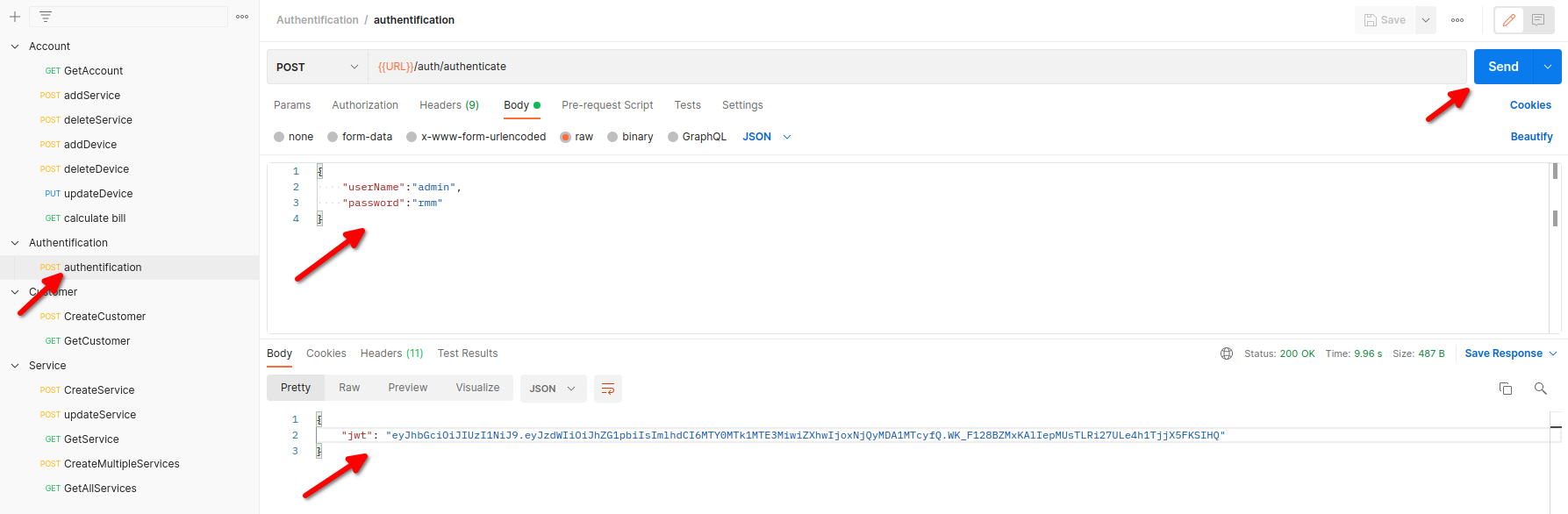
Explanation like you can run project to load test data to run the project.

Once you import postman workspase

* Run project
* Select in postman environment "local"

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/selectEnvironment.png)

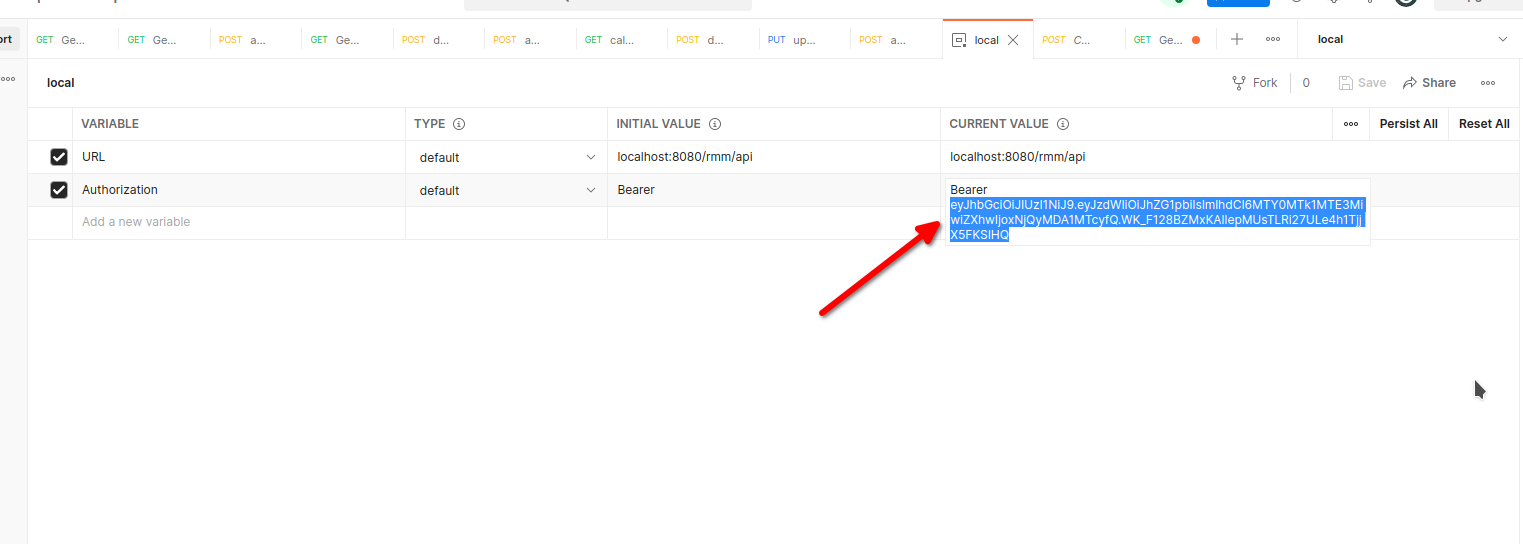
* Exists a request in the colletion "Authentification" with name "authentification", this request generate a Java web token which allow conect all request

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/requestAuth.png)

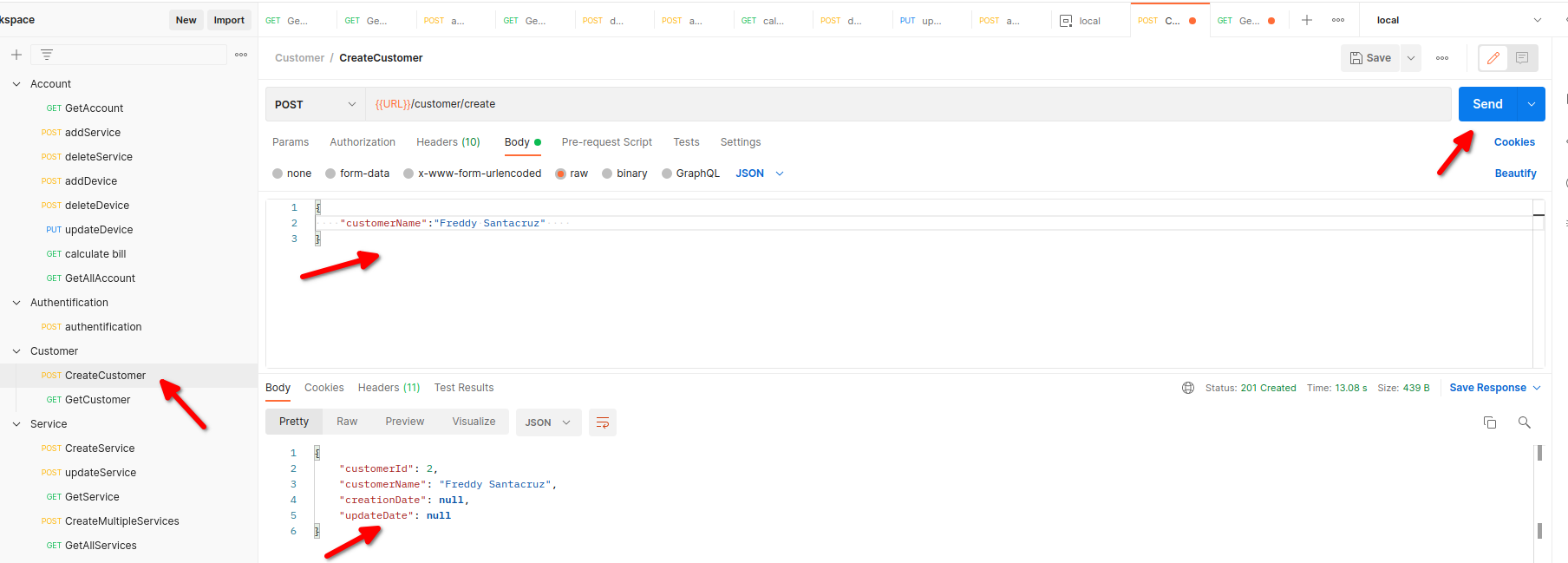
* Configurate variable "Authorization" note: copy jwt generate with Authentification request, after edit local environment in the Authorization in Current value field paste jwt.

example ->

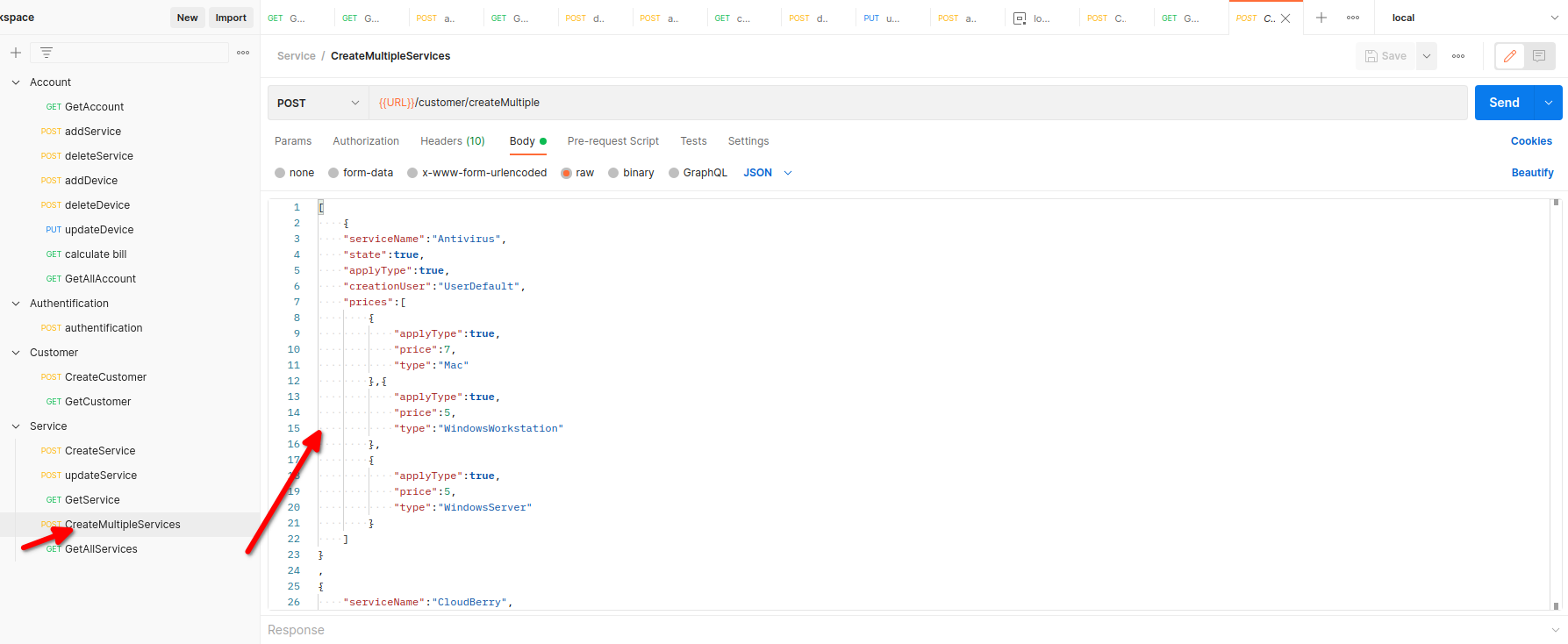
Bearer yJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJhZG1pbiIsImlhdCI6MTY0MTk1MTE3MiwiZXhwIjoxNjQyMDA1MTcyfQ.WK\_F128BZMxKAlIepMUsTLRi27ULe4h1TjjX5FKSIHQ this value should star with "Bearer ".

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/configurationEnvironment.png)

* You choose from Customer Colletion -> CreateCustomer note: this request create customer also create a new accout link with it.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/createCustomer.png)

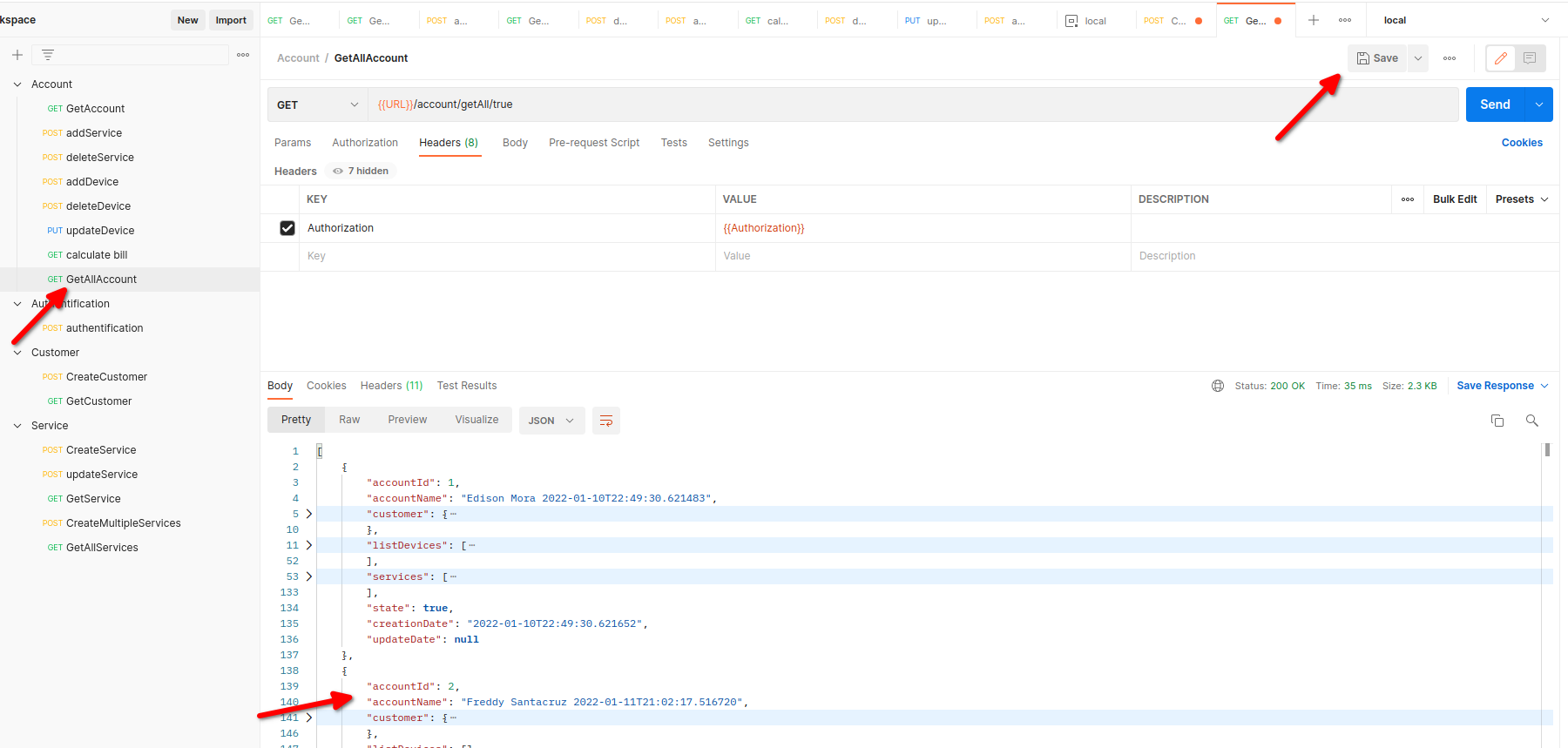
* You choose from Services Colletion -> CreateMultipleServices note: this request create services test, also link list price to service

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/createMultipleServices.png)

# Run project

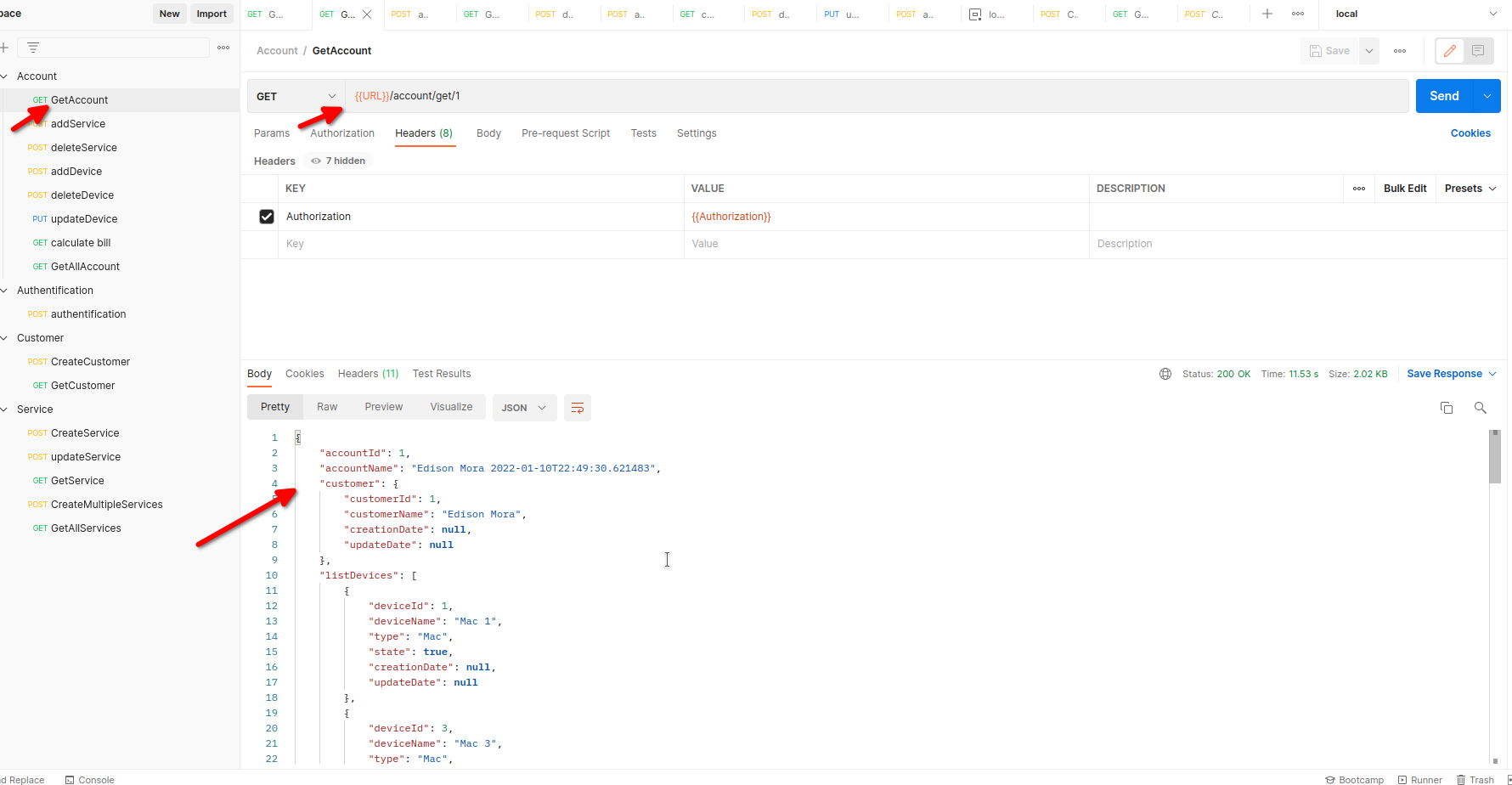
* You choose from Account Colletion -> GetAllAccount

note: this request get all accounts with state = true, copy an accountId

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/getAllAccount.png)

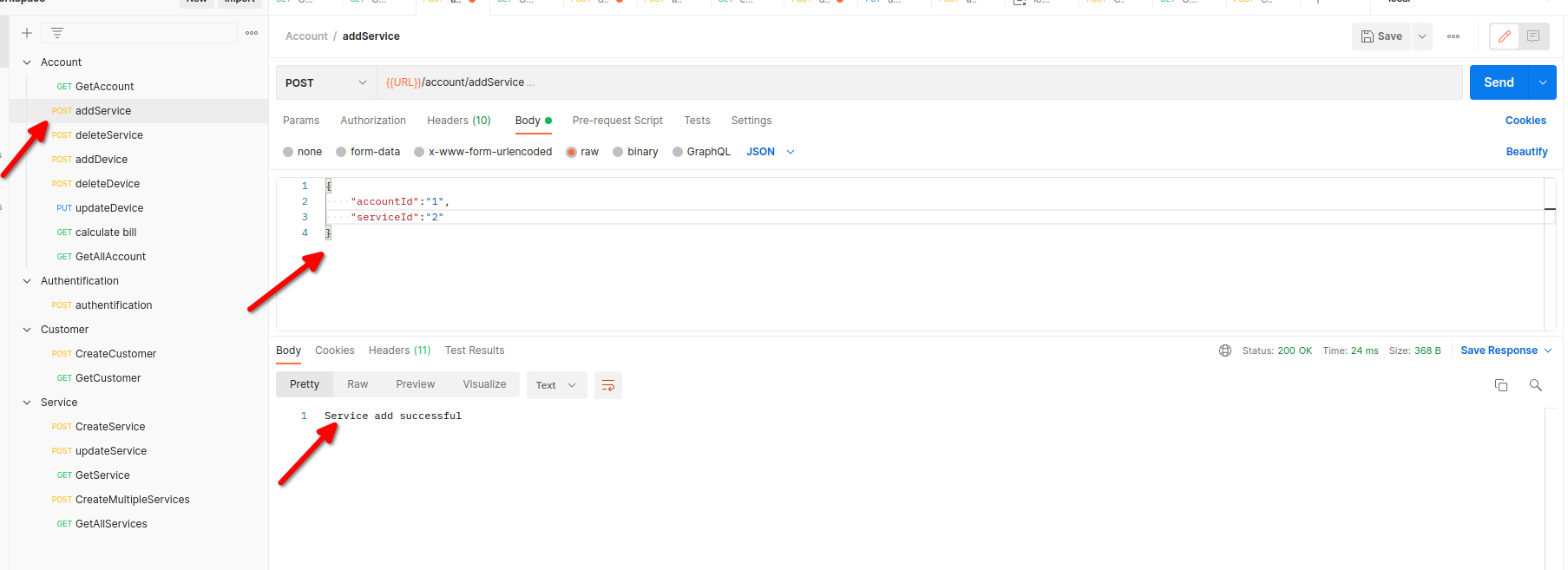
* You choose from Account Colletion -> GetAccount

note: this request get account by accountId

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/getAccount.png)

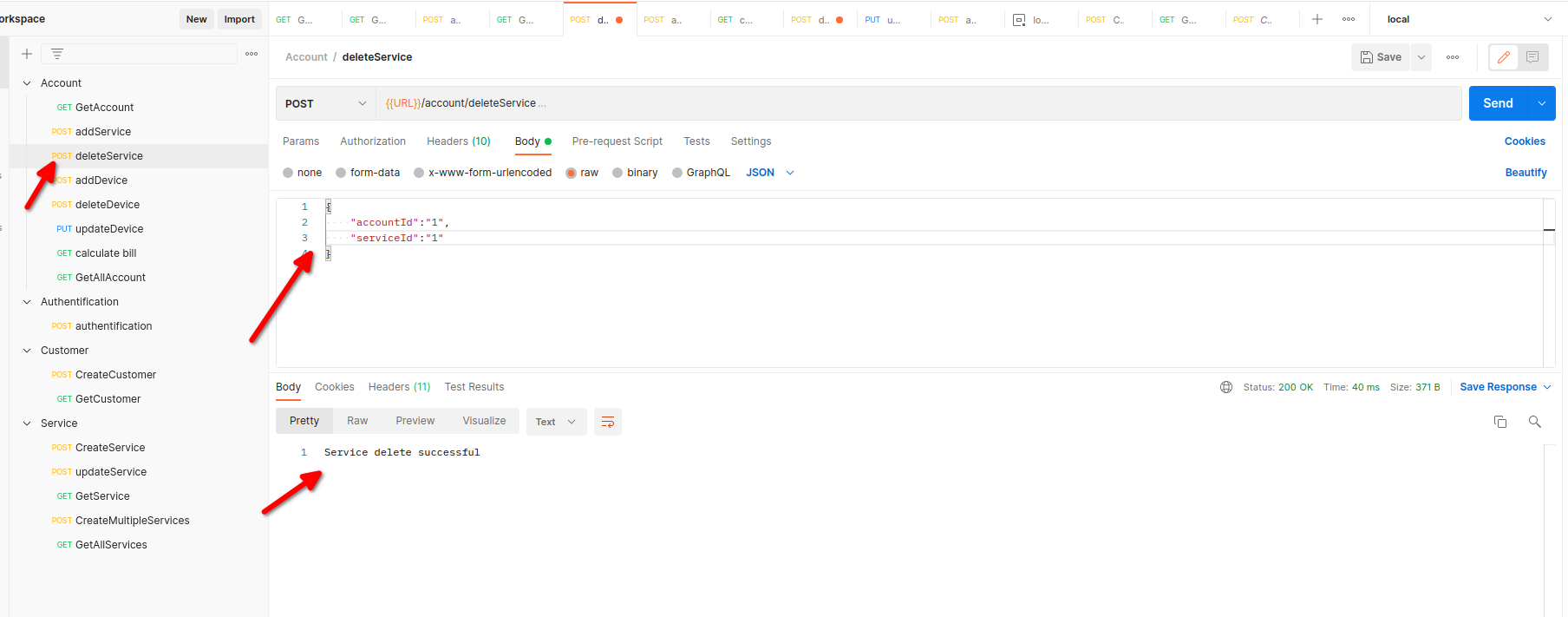
* You choose from Account Colletion -> addService

note: this request add service to account, you must send a object with accoutId and serviceId.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/addService.png)

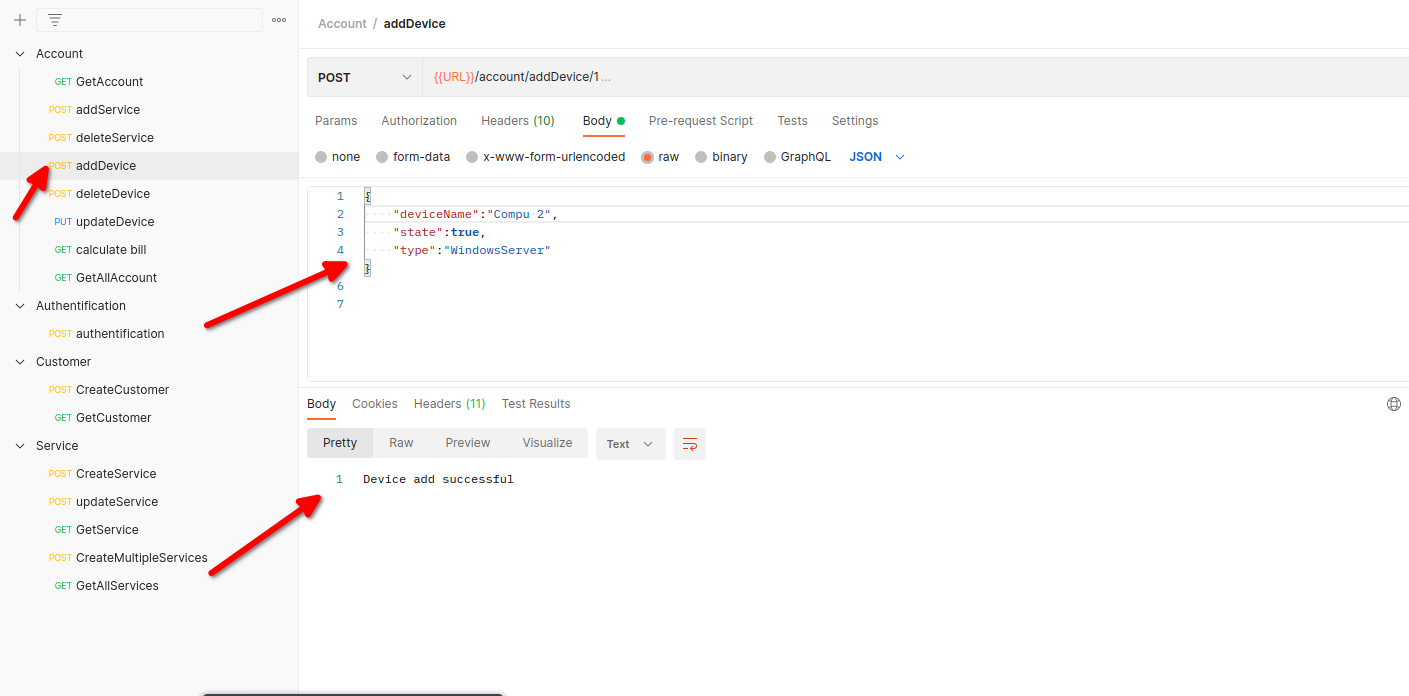
* You choose from Account Colletion -> deleteService

note: this request delete service to account, you must send a object with accoutId and serviceId. this request delete link not service.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/deleteService.png)

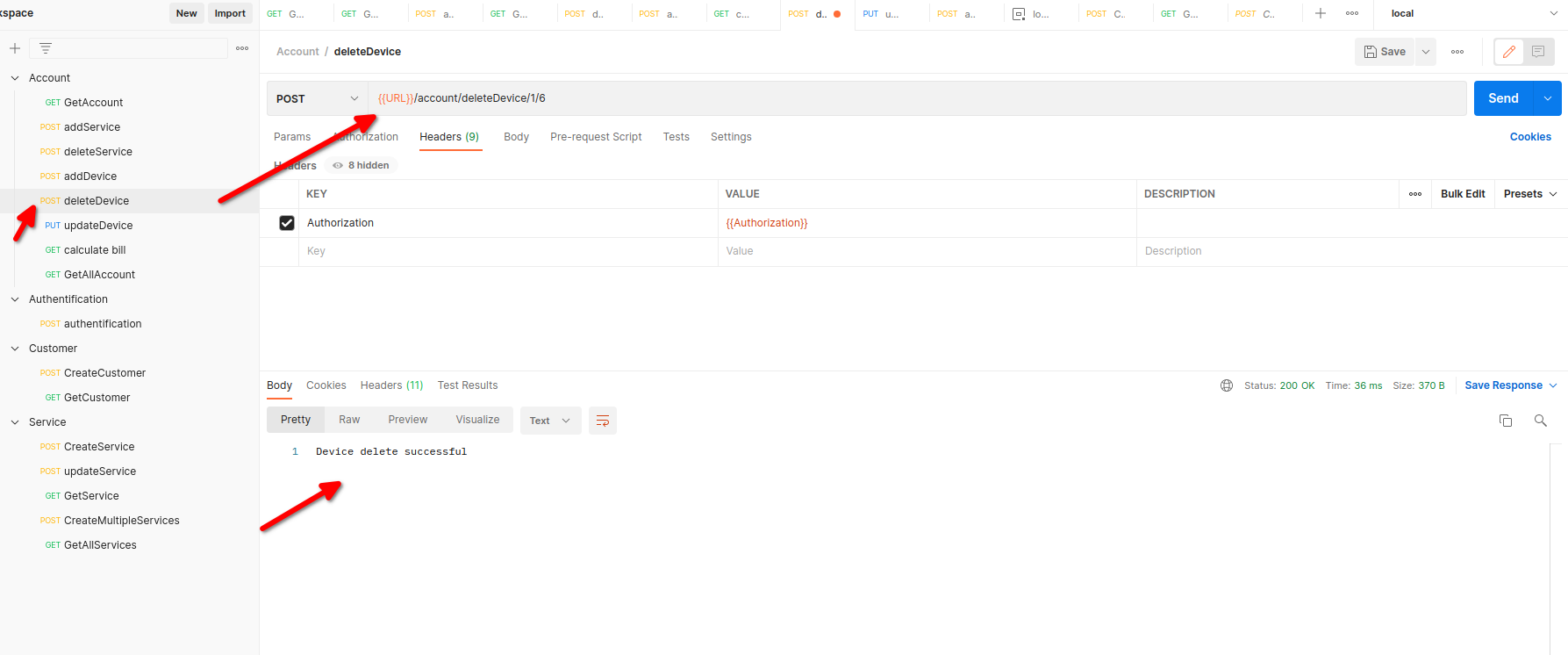
* You choose from Account Colletion -> addDevice

note: this request add device to accout, you must send accountId over URL and an Object with device data.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/addDevice.png)

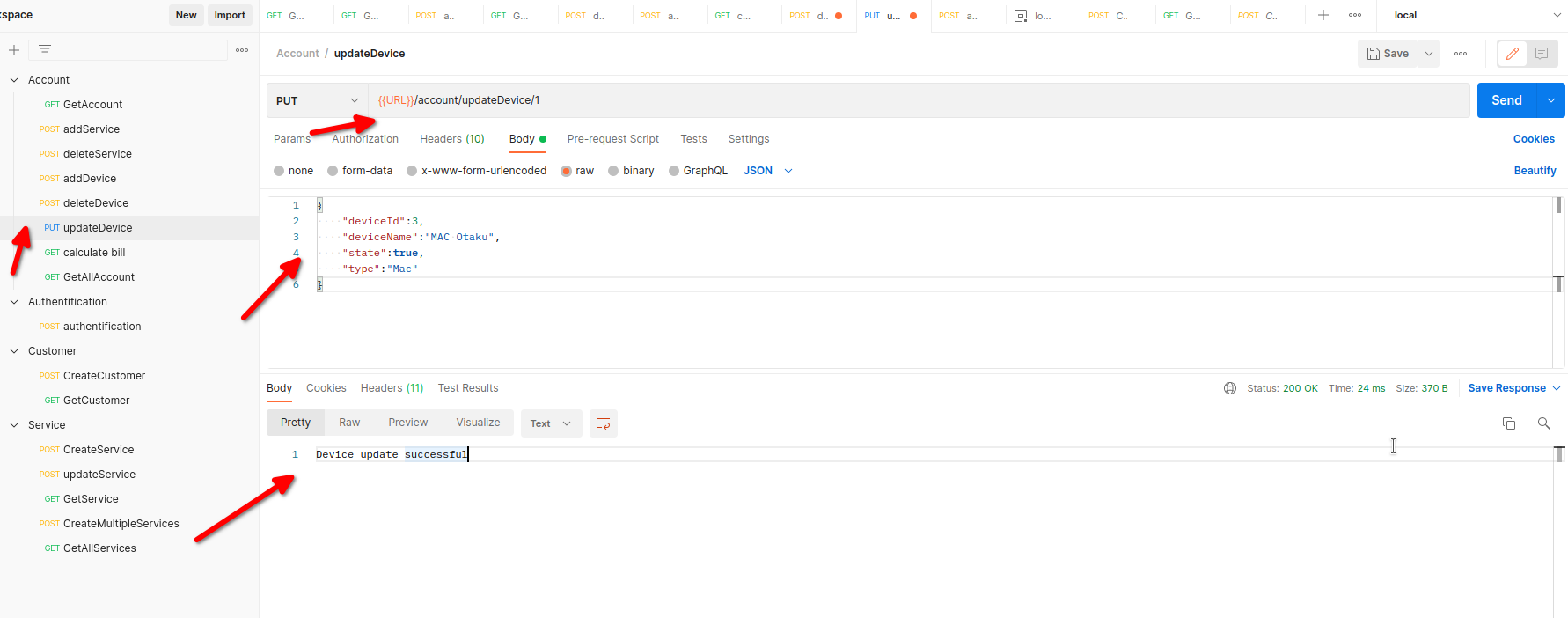
* You choose from Account Colletion -> deleteDevice

note: this request delete device, you must send accoutId and deviceId over URL.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/deleteDevice.png)

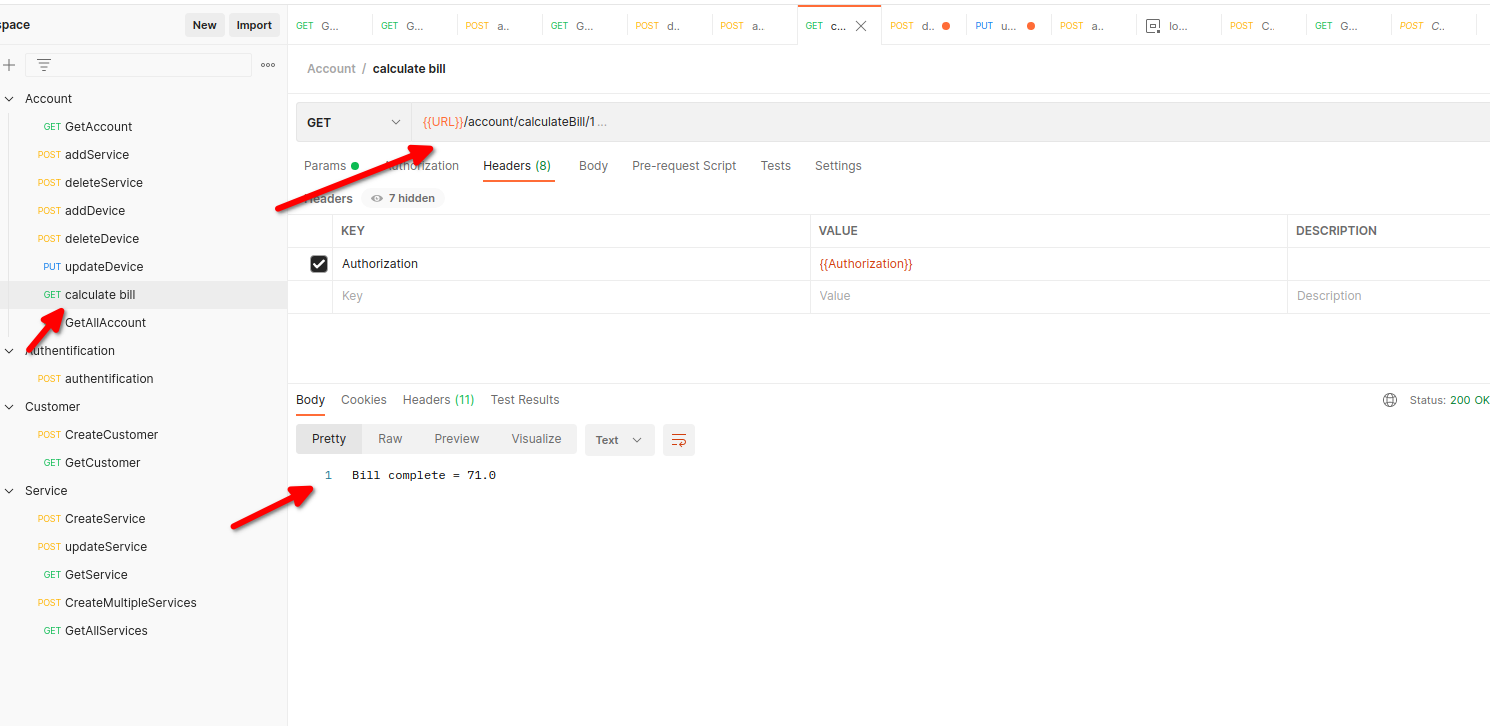
* You choose from Account Colletion -> UpdateDevice

note: this request update device, you must send deviceId over URL.

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/updateDevice.png)

* You choose from Account Colletion -> calculate Bill

note: this request calculate total bill by accountId

[](https://github.com/santacruzfreddy/rmm-services-server-app/blob/master/images/calculateBill.png)

# Test JUnit and Mockito

In this project I did some test bellow breakDown

* test calculate bill
* test not exist account
* test not exist service
* test delete\_service\_error\_when\_null\_parameter

configurate mockito data to testing

# JWT security APi

For security I used JWT (java web token) to create token with 3 hours expired time.

* User -> admin
* password -> rmm

Thanks for read

Copyrigth Freddy Santacruz