**CONTENT**

1. Creating Test plans with threads , sampler, listeners ------------------------

2. Performance Testing with CLI Mode -----------------------------------------

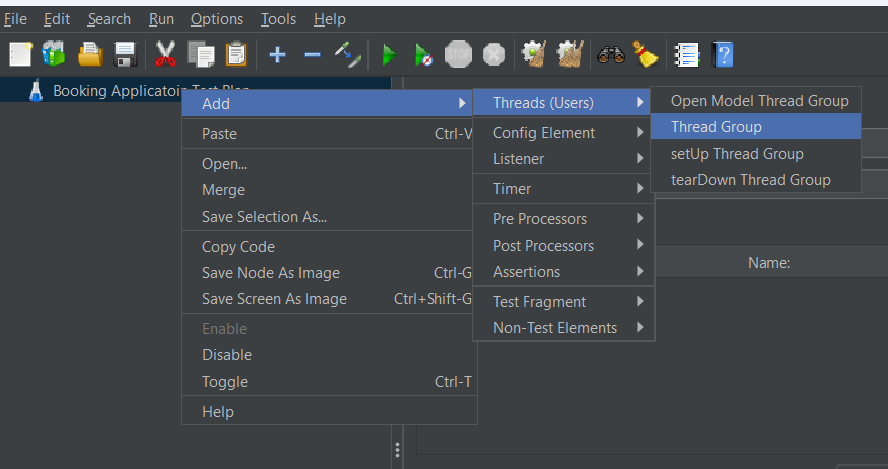
**# Booking Api Project:**

Open JMeter: Open JMeter folder – go to bin file > double click on jmeter.bat

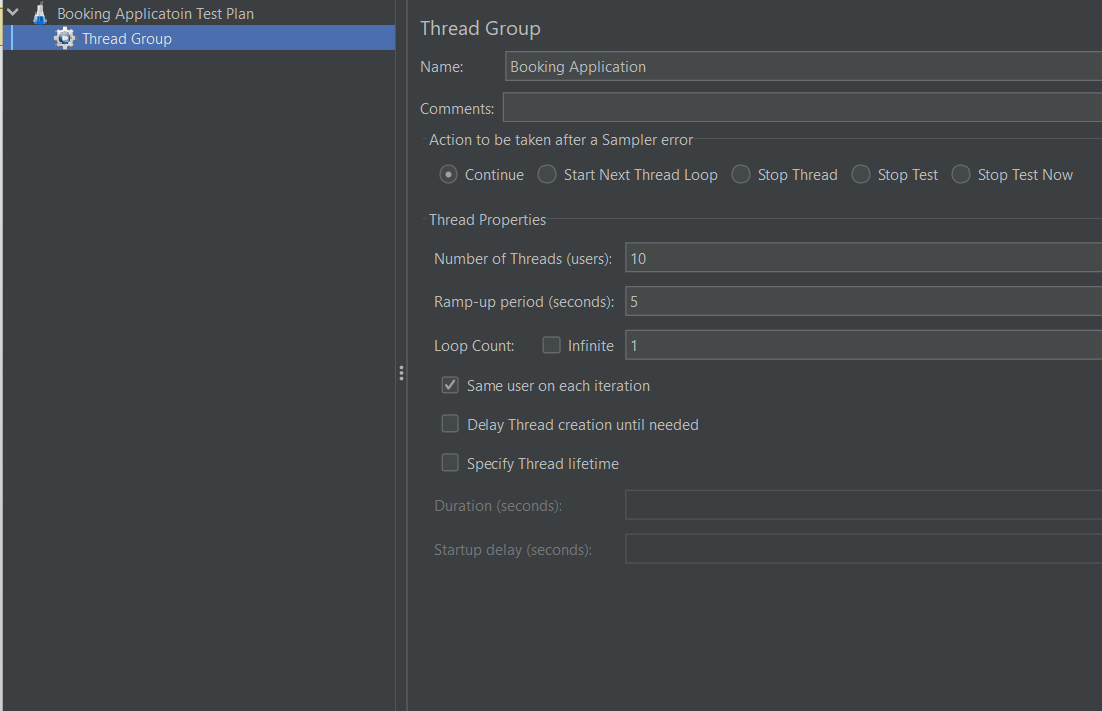
* **Creating Project on JMeter:** Add Threads – Sampler – Listern ( to show the result and add always in last for the final result show)
* Documentation URL: [http://restful-booker.herokuapp.com/apidoc/](http://restful-booker.herokuapp.com/apidoc/%23api-Auth)

1. **Threads Group** – No of users use the apps.

* **Process:** right click on test plan – add – threads – thread Group. After that rename (Booking Application) according your project.



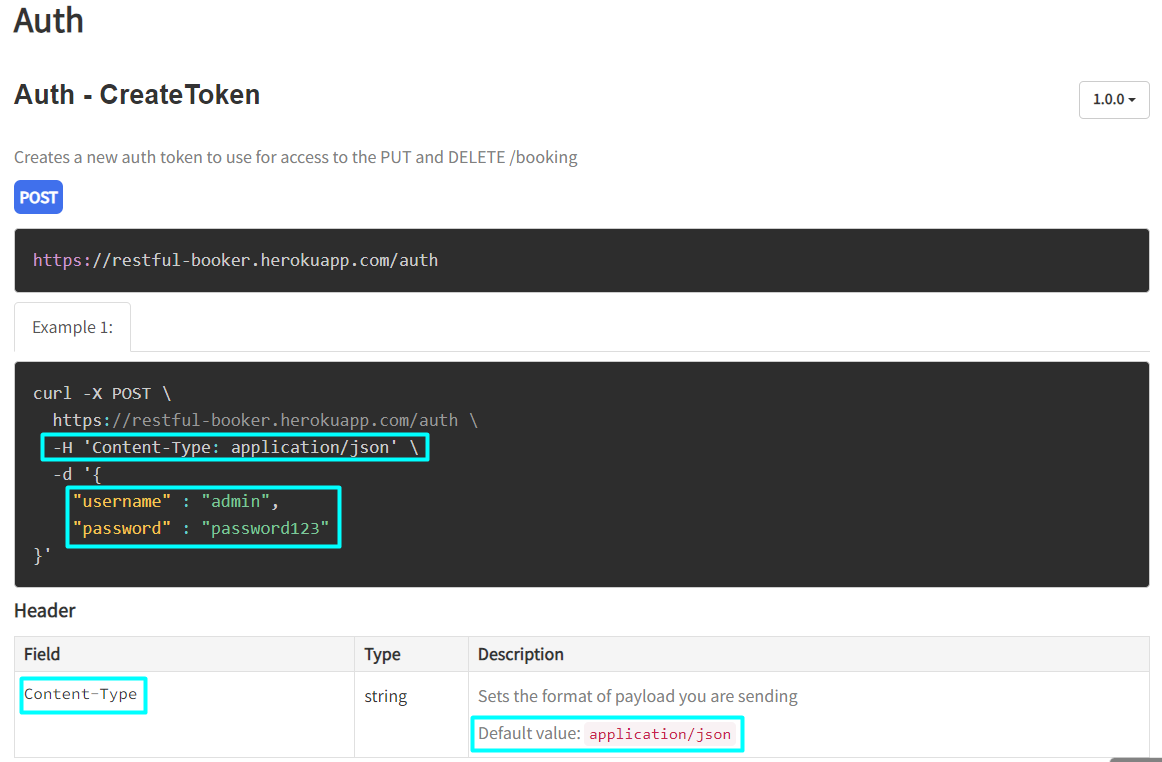
After adding thread group now, we need to add no of users and ramp-up periods (how long to take to "ramp-up" to the full number of threads/users)



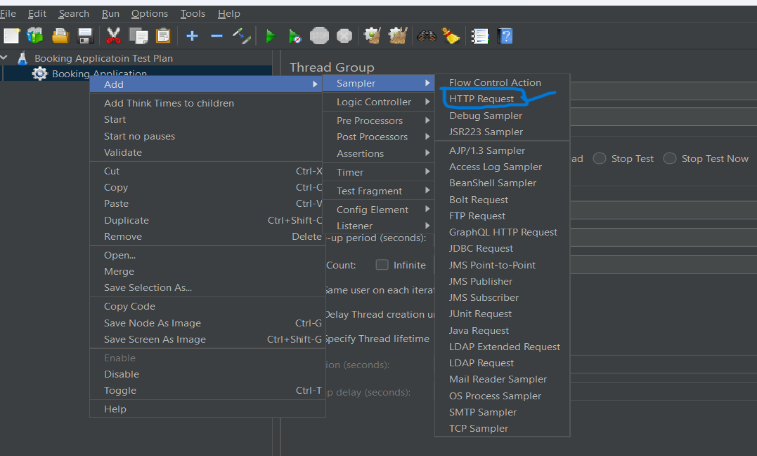
1. **Sampler Add** – the actual requests which are sent to the server.

**2.1** Doing for Login. Do as it is for other’s following documentation by data.

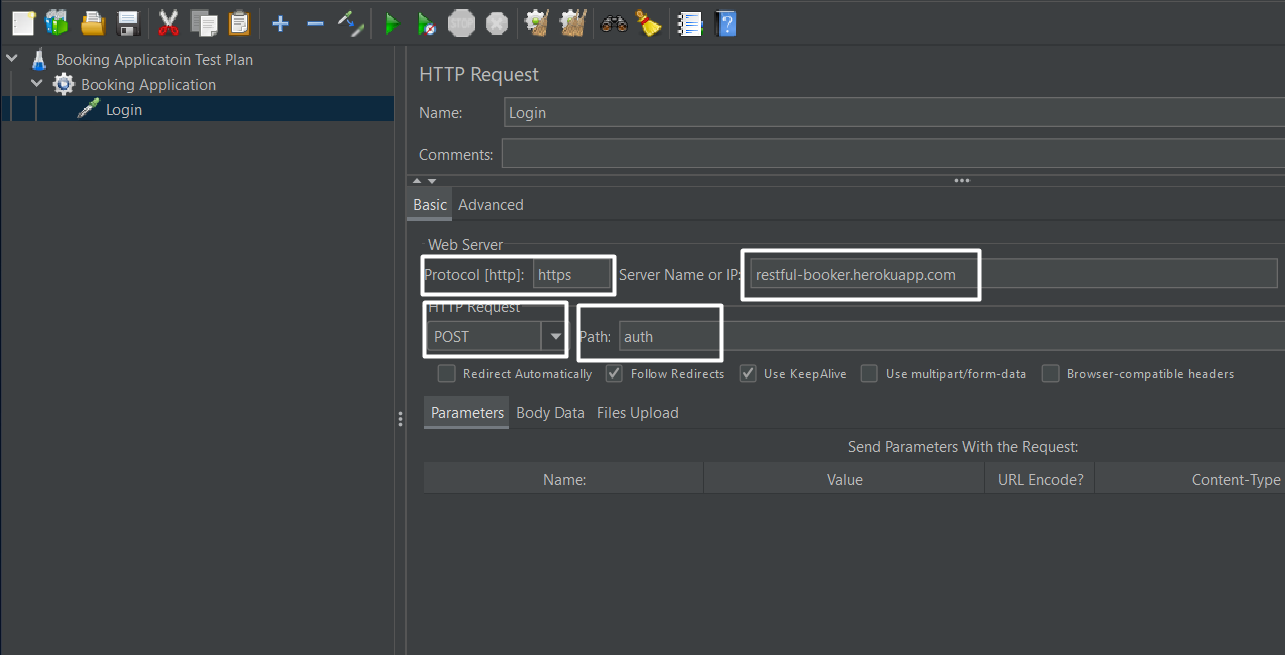
* **Given Documentation**:



* **Process:** right click on threads group – add – sampler – select according to the project. There are many samplers but we use HTTP Request for our project and rename (Login) according to it.



Not it’s time to add api which need to be done very carefully. Our Login Method is POST and api - <https://restful-booker.herokuapp.com/auth>.

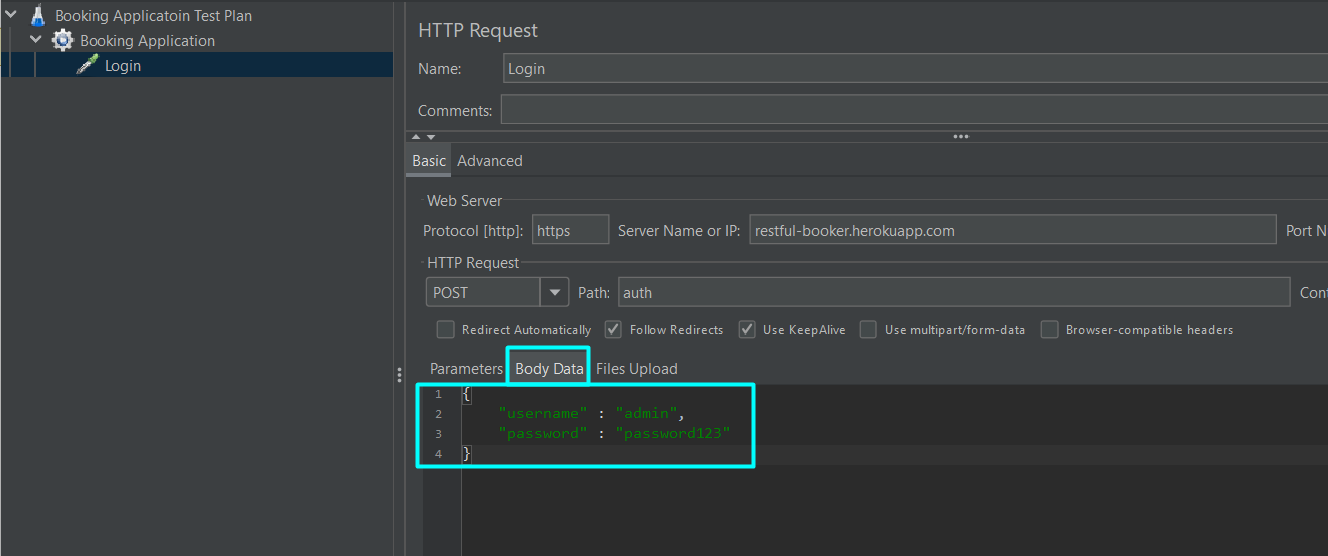


Below All are placed as given in documentation. [Note no need to add **//** or **:** or ; ]

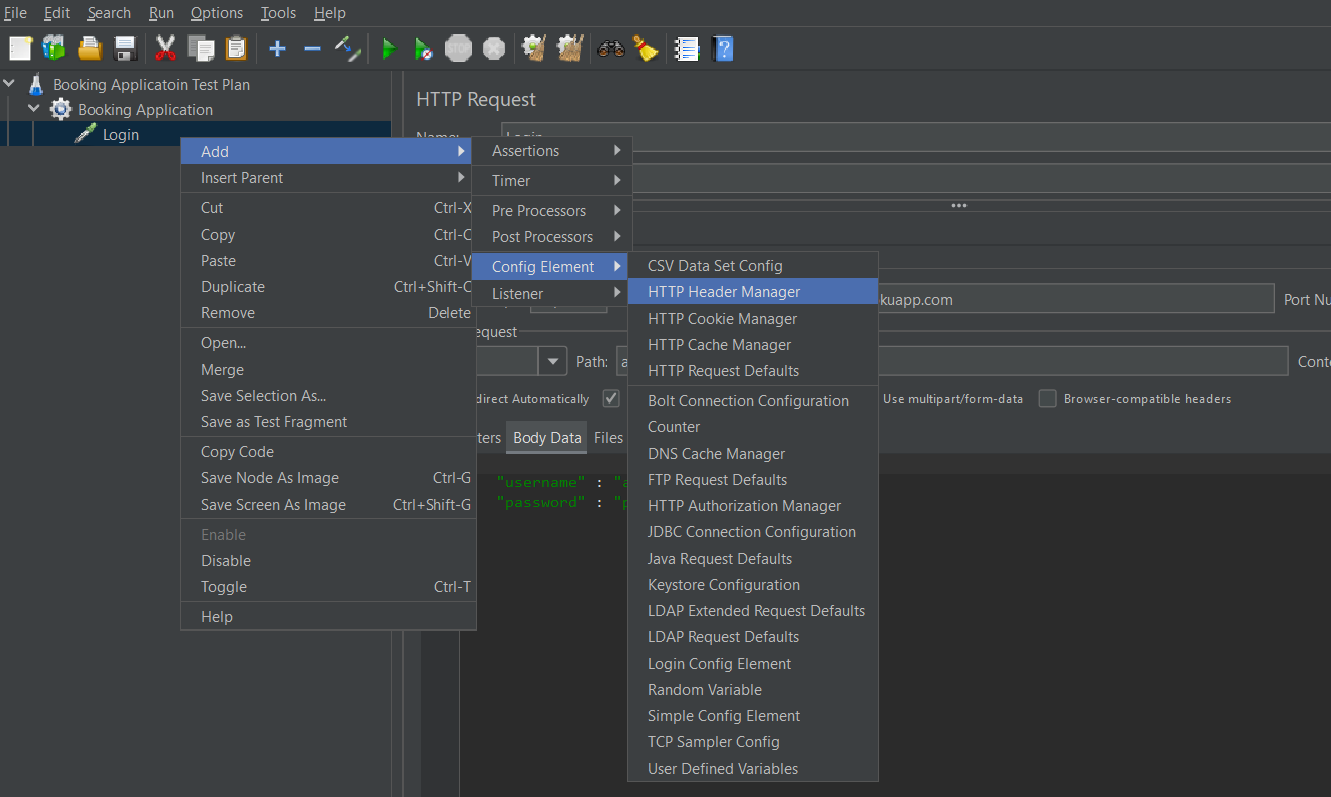
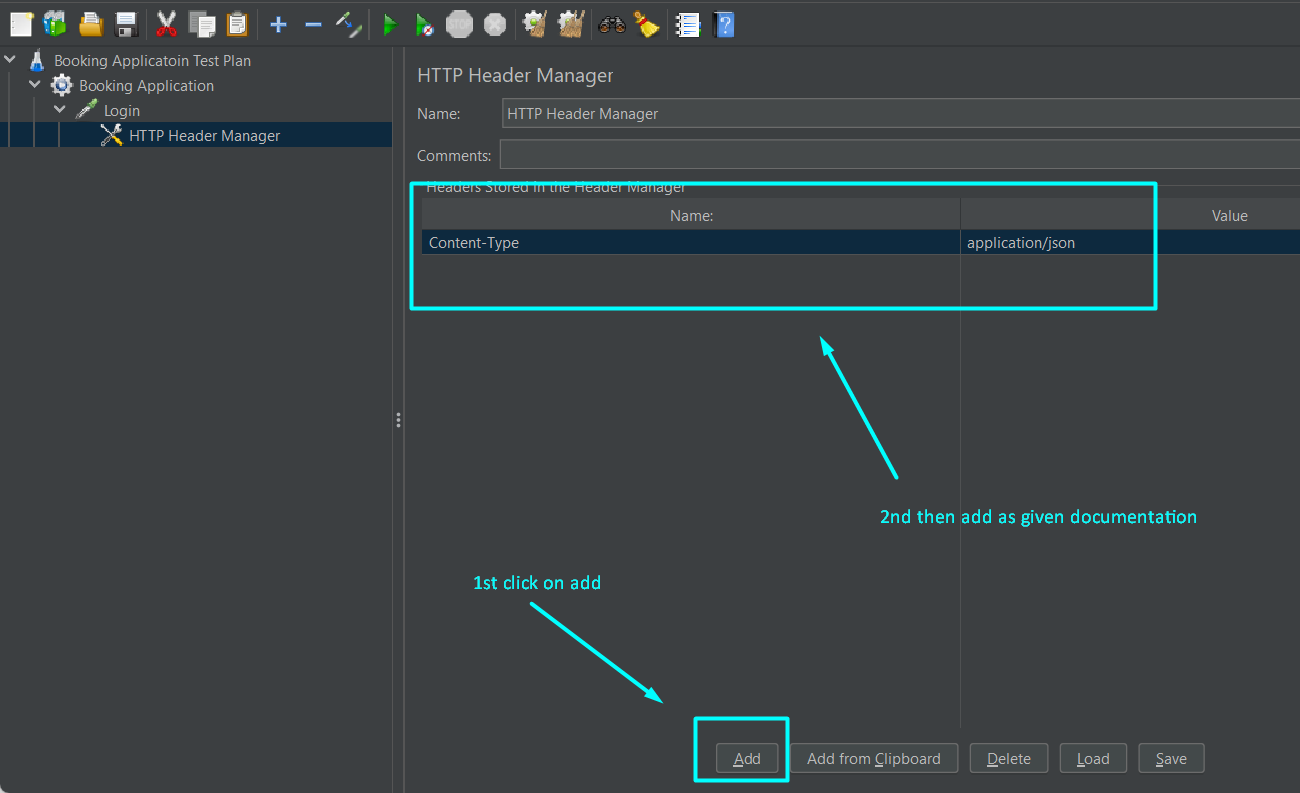
* On Protocol Box: **https / http**
* On Server Name Box: **restful-booker.herokuapp.com**
* On Port Box: (Everything after.com) here only **auth** left that’s why auth
* On HTTP Request Box: **POST** Method

Now Add given data and header in JMeter sampler section.

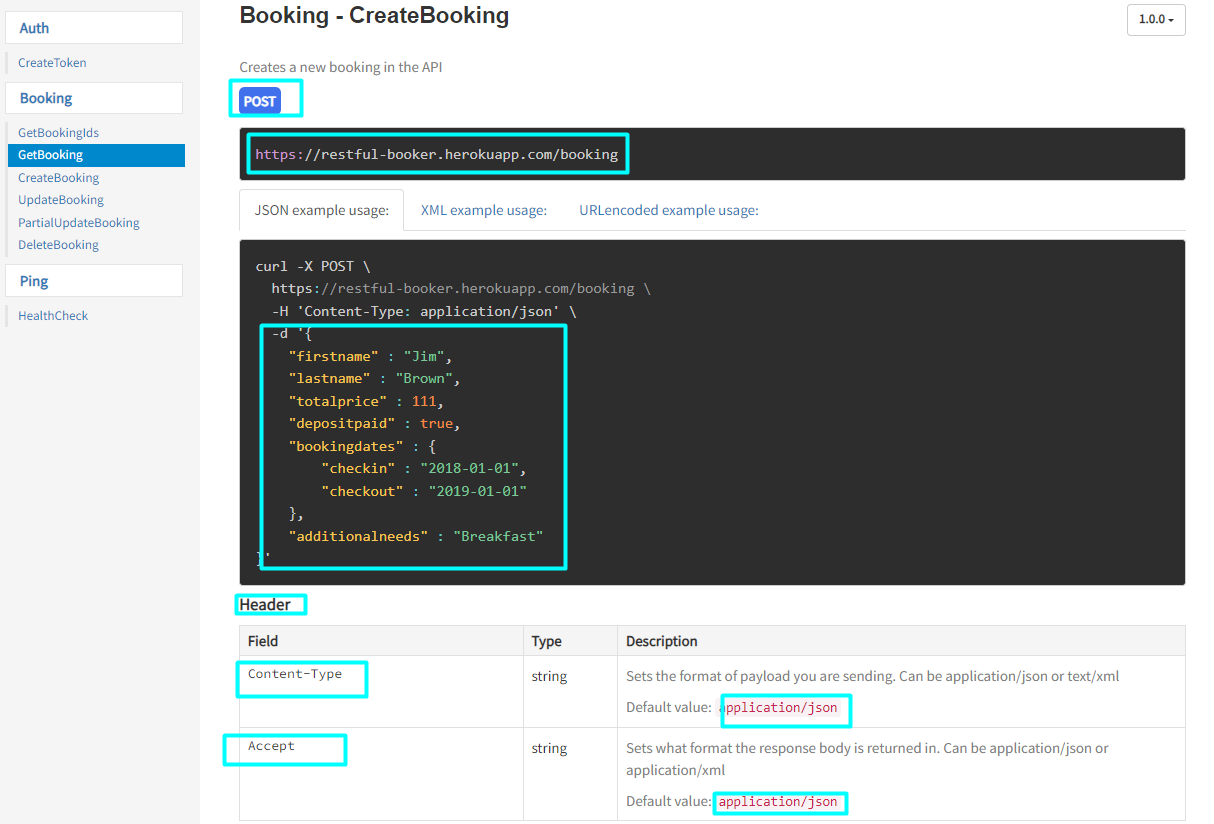
* **Body data:**



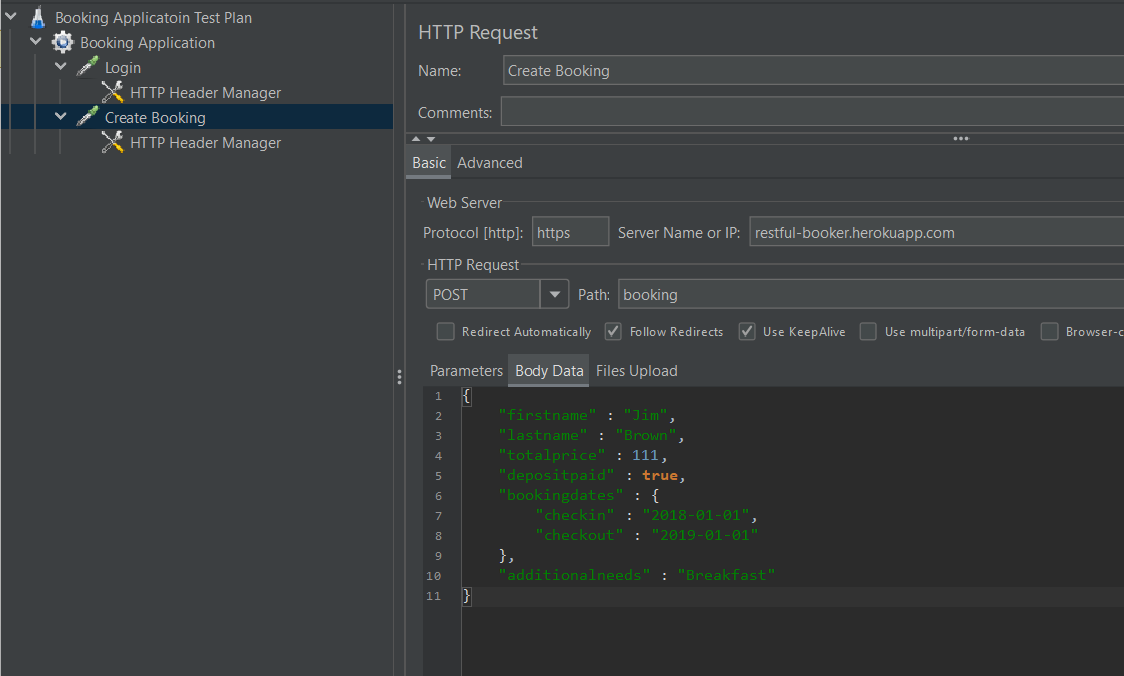
* **Header Element:** 
  + H 'Content-Type: application/json'

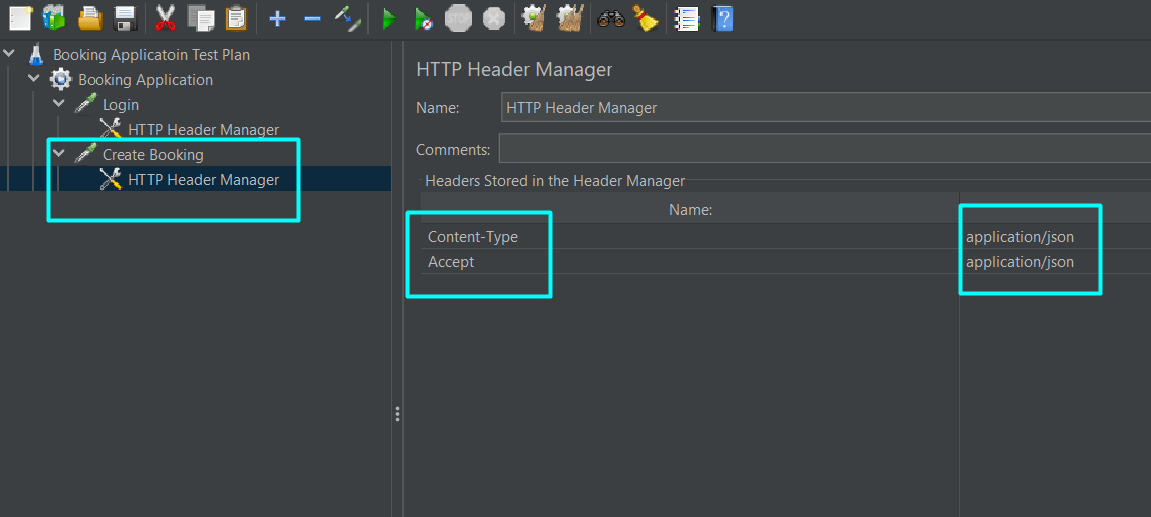
 

* 1. Doing for Create Booking –
* **Given Documentation**:
* **API -** https://restful-booker.herokuapp.com/booking
* **Method** – POST
* **Header** – Content-Type & Accept – application/json

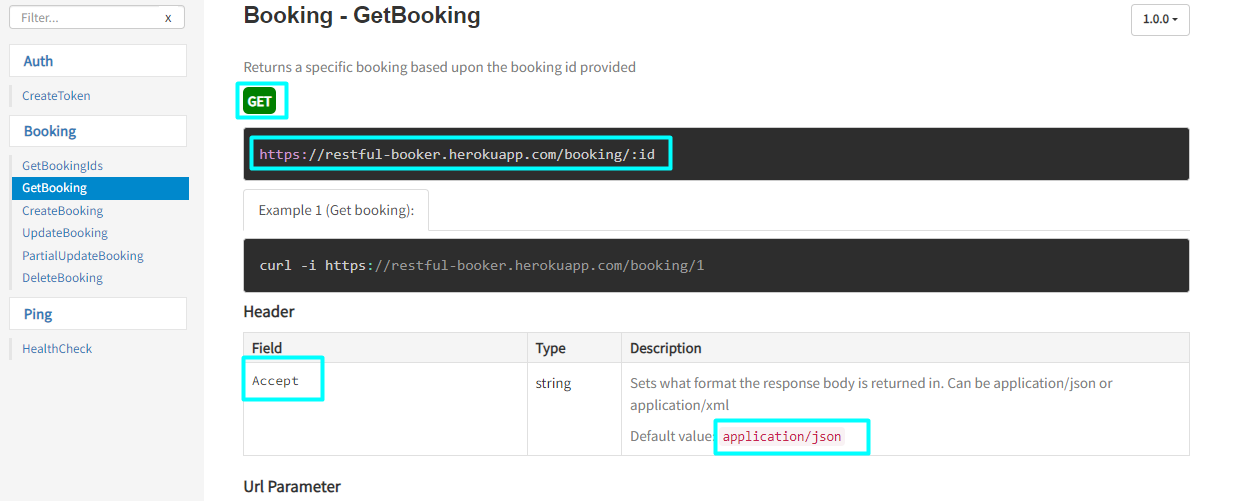


* **Process and Header add and Body Data Add:** Same As 2.1





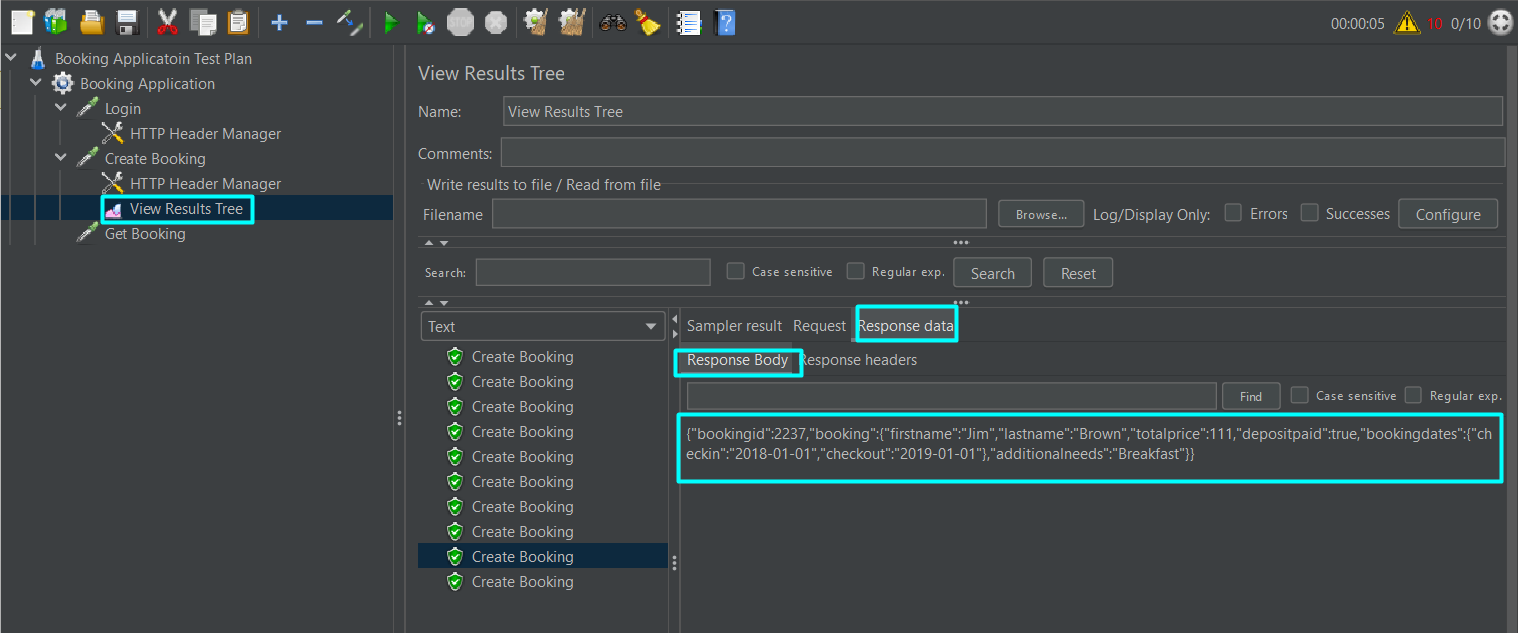
* 1. Doing for Get Booking –
* **Given Documentation**:
* **API -** https://restful-booker.herokuapp.com/booking**/:id**
* **Method** – GET
* **Header** –Accept – application/json
* [Note – Here id in api is dynamic. So, we can’t get booking without id and for that we need to add id as dynamic id.]



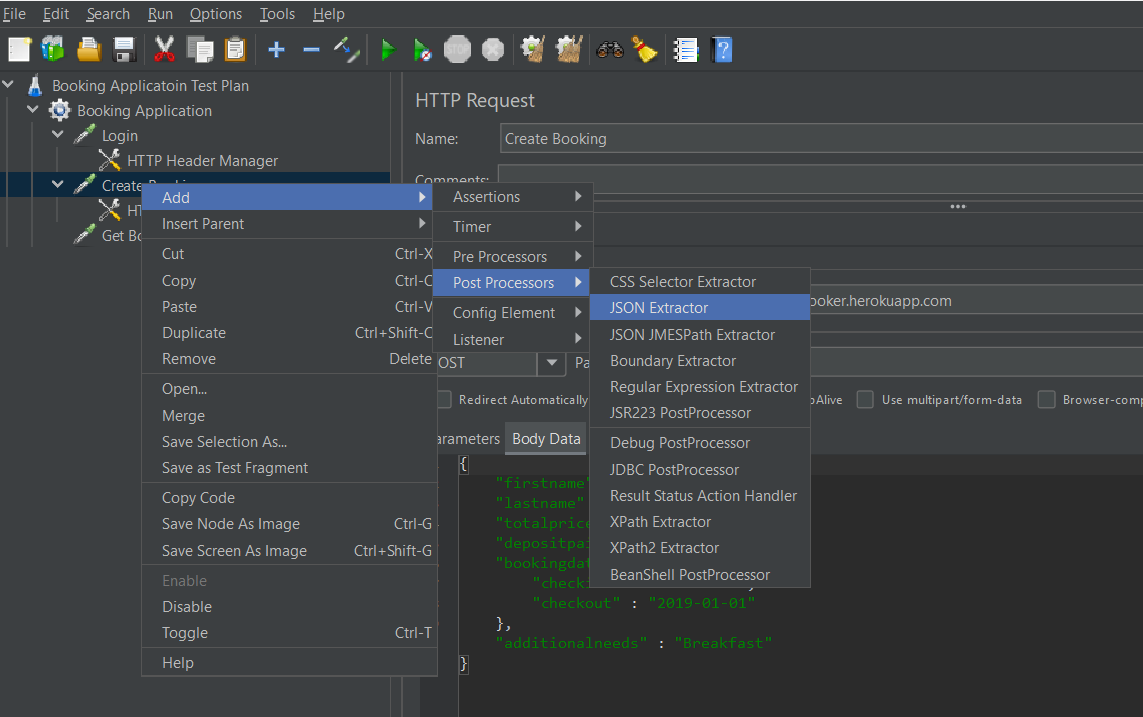
**2.3.1 First do dynamic id.**

Since id is generating from **Create booking**, we need to add **JSON Extractor** on create booking.

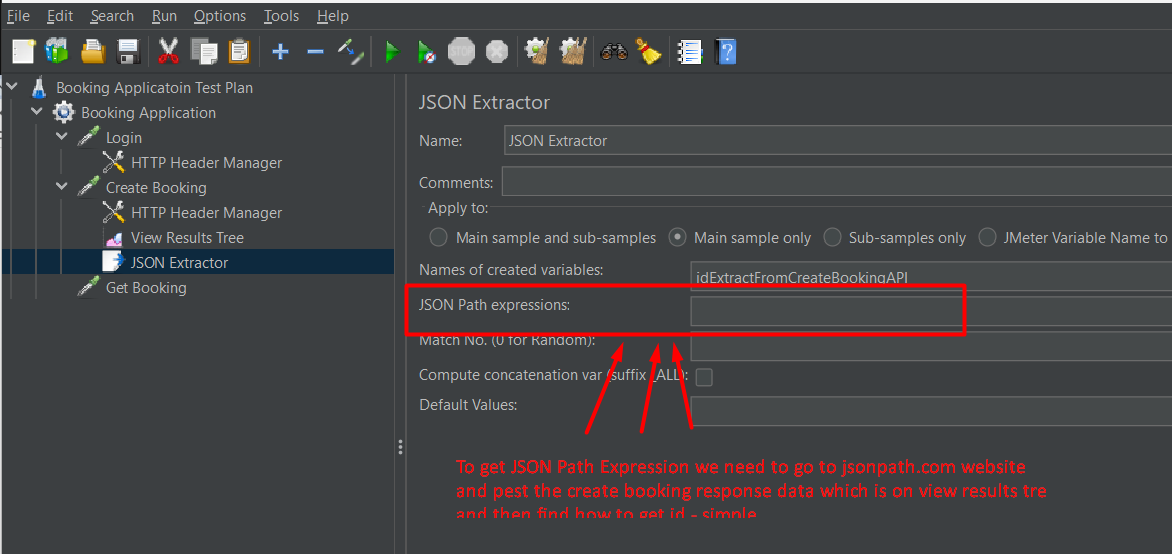
* 1st add a **view results tree** – right click on create booking – add – listener – view results tree. And run once the project.



* 2nd for **json extractor**, Right click on create booking – add – post processors-JSON Extractor.

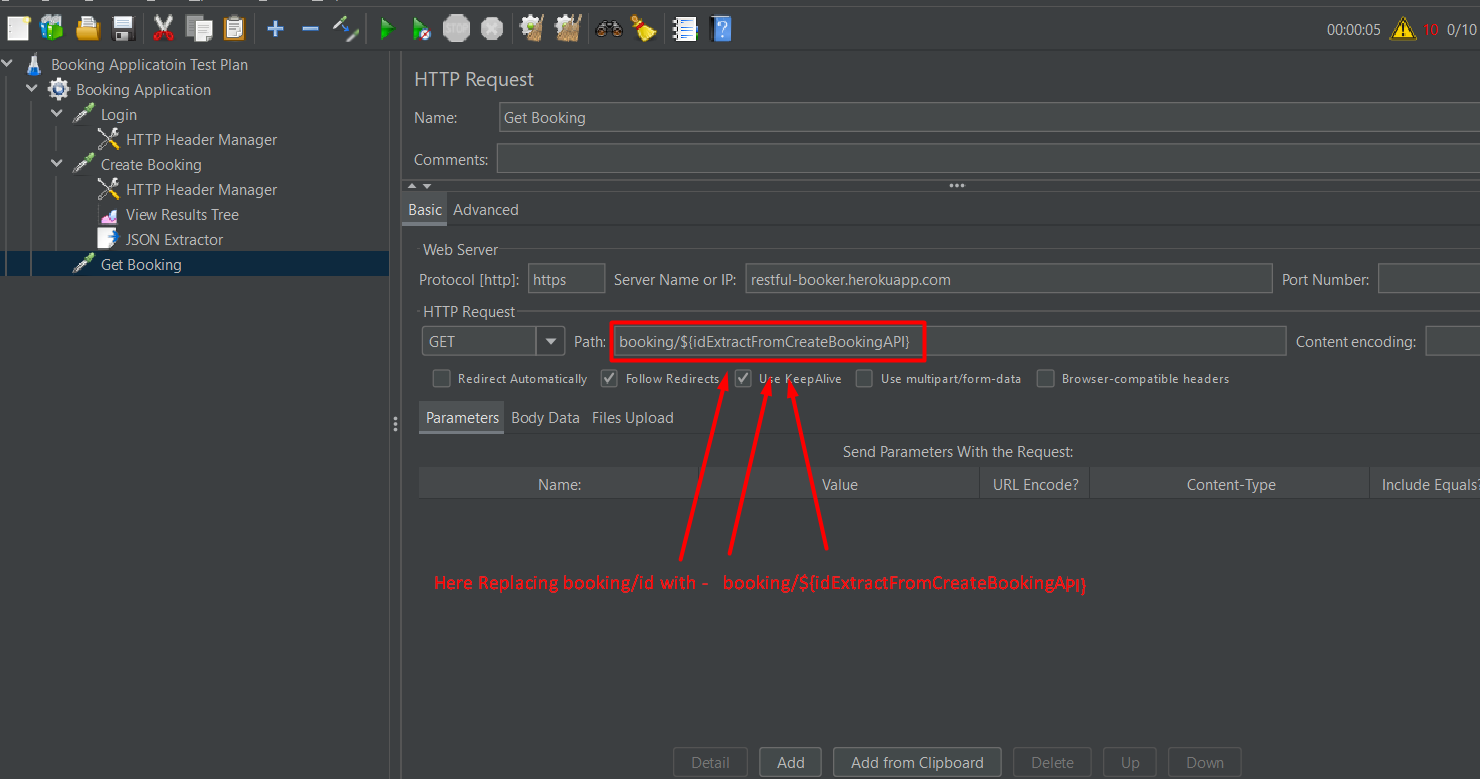


* Give a suitable name in – “**Name of created variables**” filed
* For JSON Path Expresion – Go <https://jsonpath.com/>



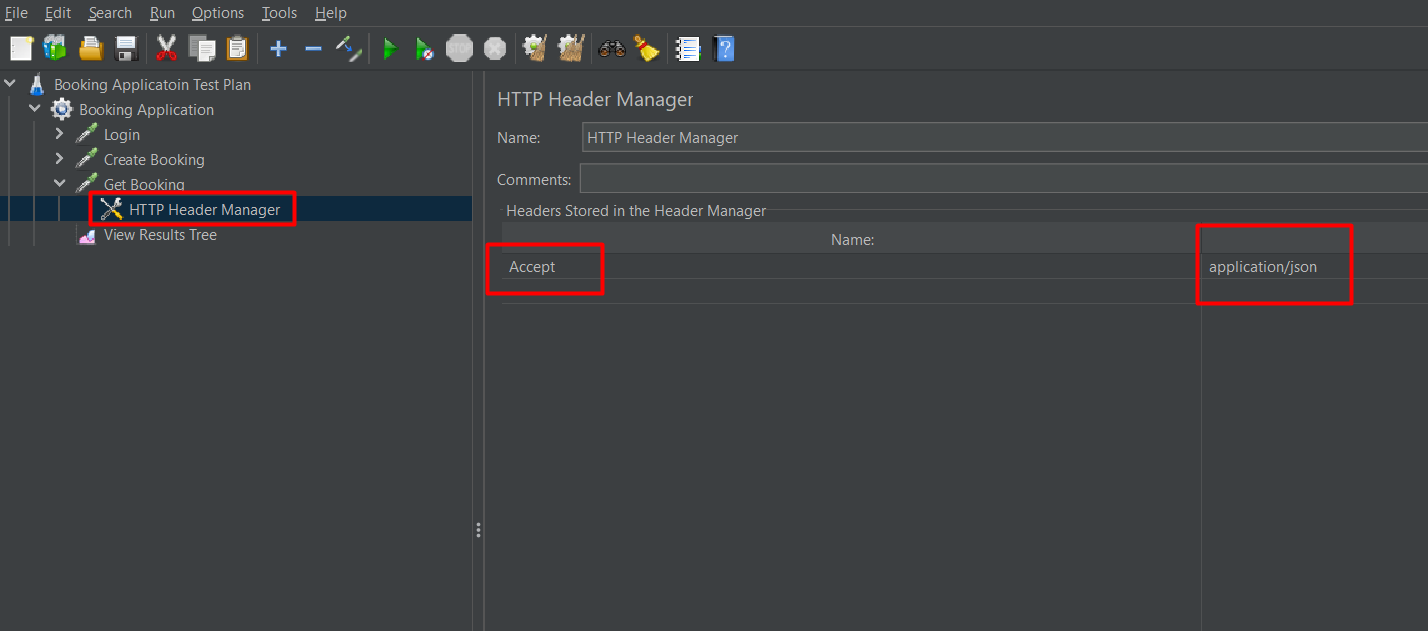
* Now copy the view result tree body data from create booking and pest on JSON path website and find path for ID. Which is – “**$.bookingid”** and now paste it on JSON path Expression.

**2.3.2 Process for Get Bookings:** Process and Header add and Body Data Add by following Same As 2.1 just replace as given documentation.

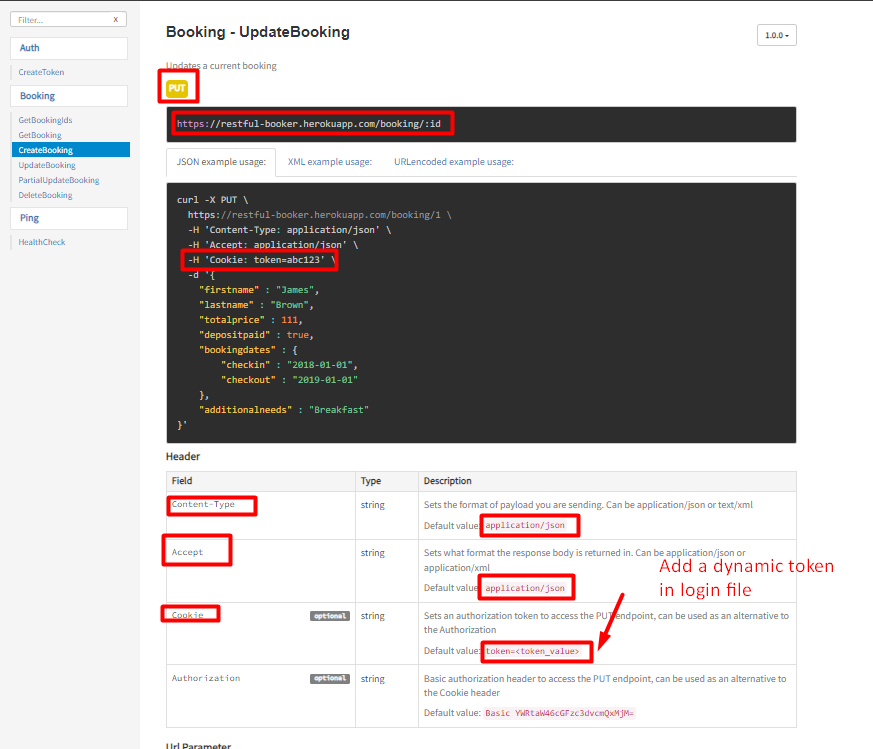


Given api - https://restful-booker.herokuapp.com/booking**/:id,** we replace **:id** with

**${idExtractFromCreateBookingAPI}** which we already get from Create booking Json Path file (see 2.3.1). Also add **Header manager** given as documentation. See below



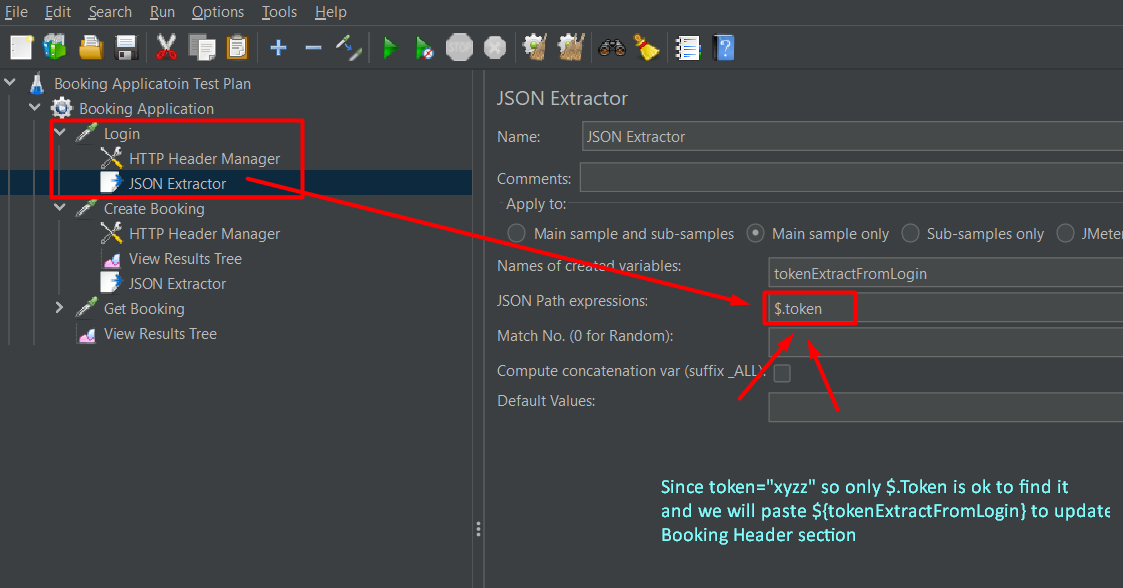
* 1. Doing for Update Booking –
* **Given Documentation**:
* **API -** https://restful-booker.herokuapp.com/booking/:id
* **Method** – PUT
* **Header** – Content-Type & Accept – application/json also need cookie – which will be need to generated dynamically in login JSON Extractor file



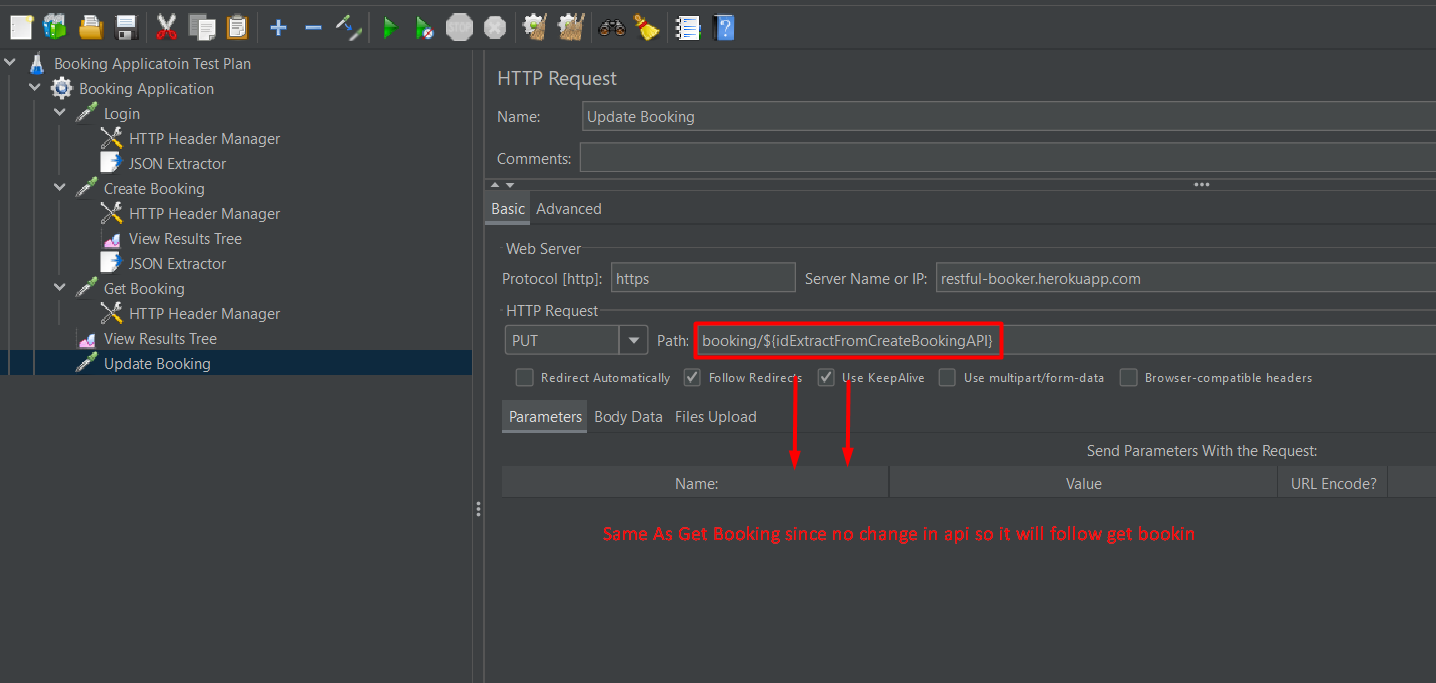
**2.4.1 First do Token dynamic.**

Since toke is generating from **Login**, we need to add **JSON Extractor** on Login.

* for **json extractor** file, Right click on Login – add – post processors - JSON Extractor.
* Give a suitable name in – “**Name of created variables**” filed here (tokenExtracFromLogin)

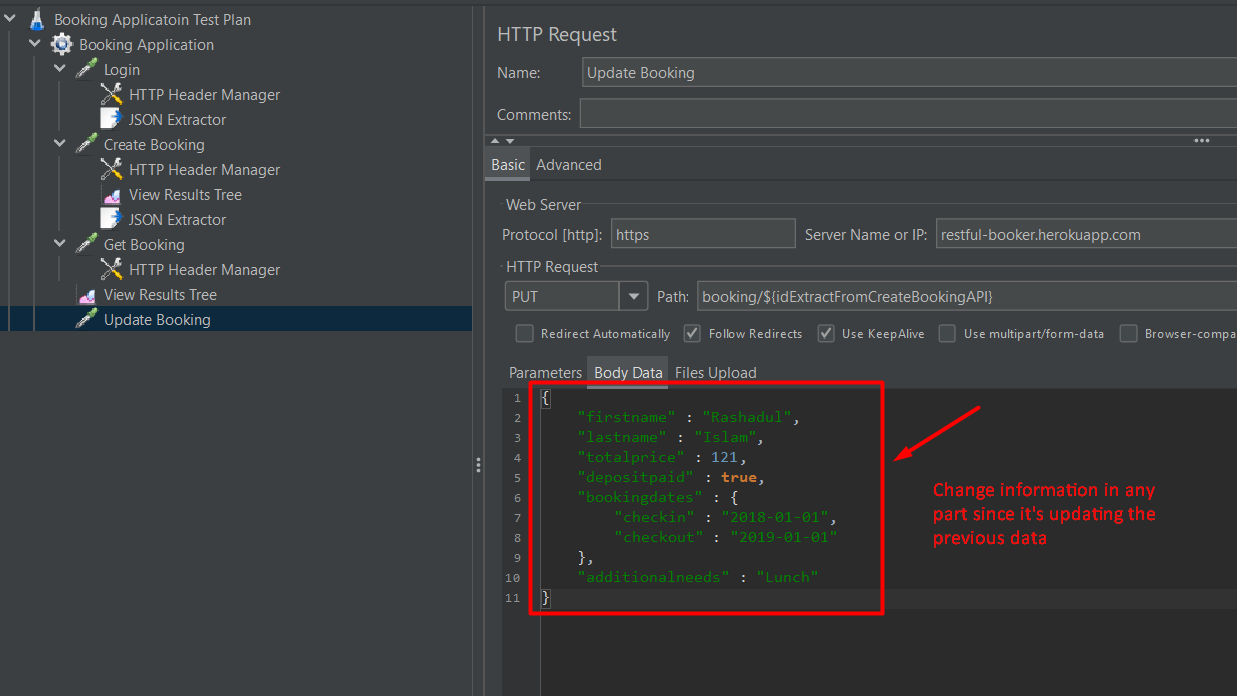


**2.4.2 Process for Update Booking:** Process and Header add and Body Data Add by following Same As 2.1 just replace as given documentation.



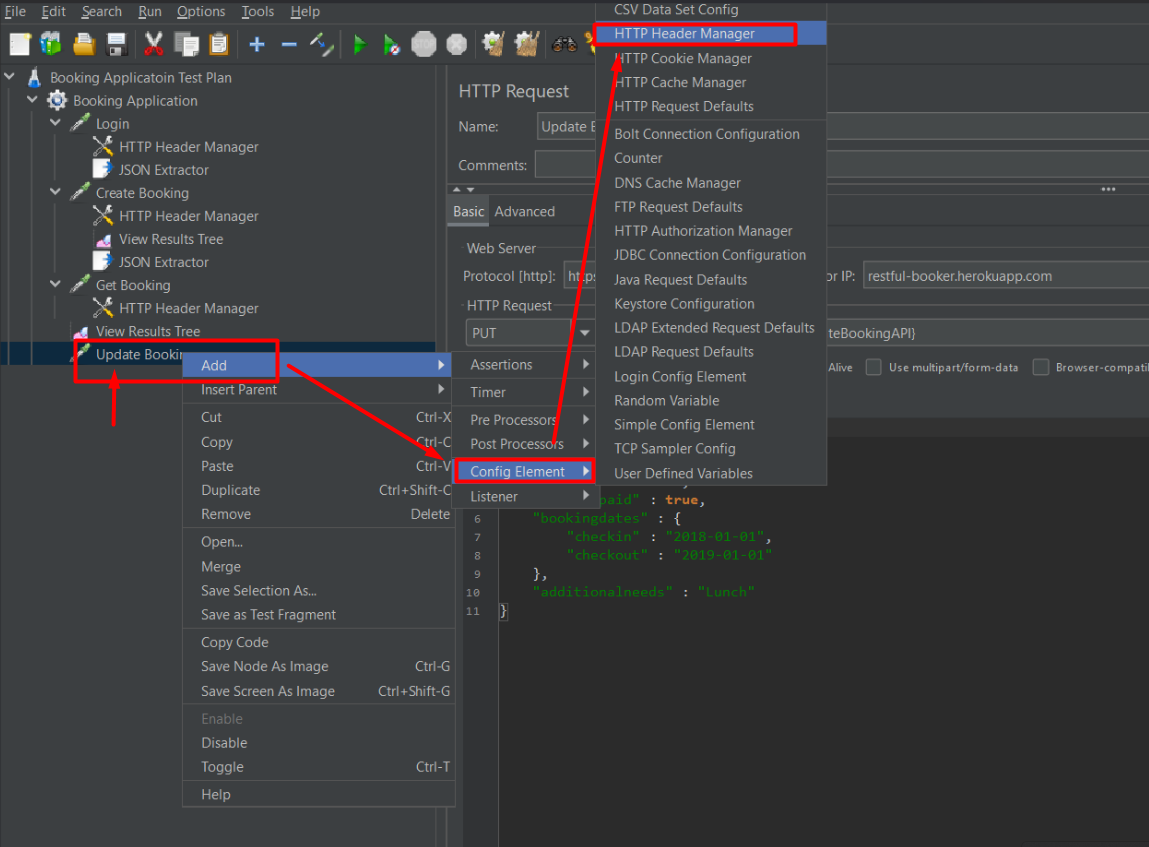
Now Add given data and header in JMeter sampler section.

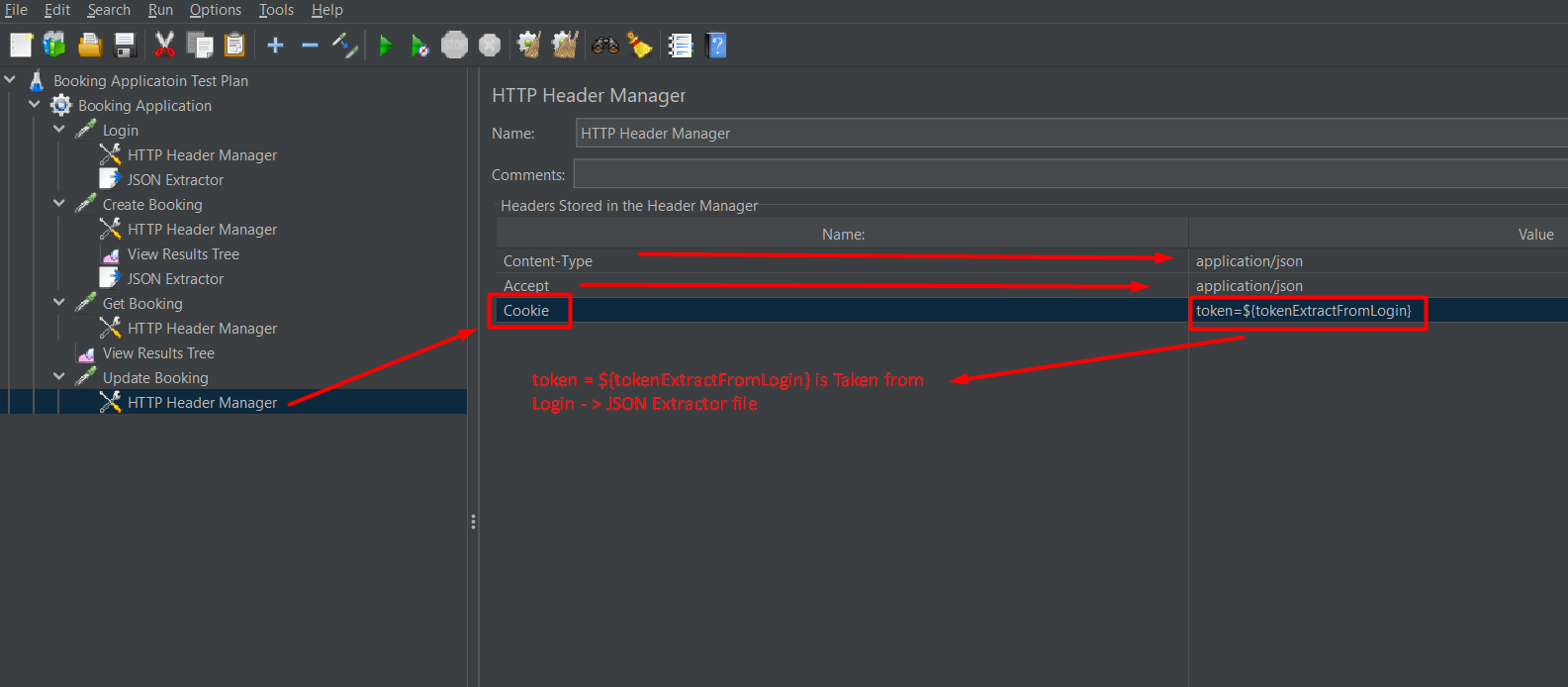
* **Body data:**



* **Header Element:** 
  + H 'Content-Type: application/json'
  + H 'Accept: application/json'
  + H 'Cookie: token=abc123' it will be replaced with

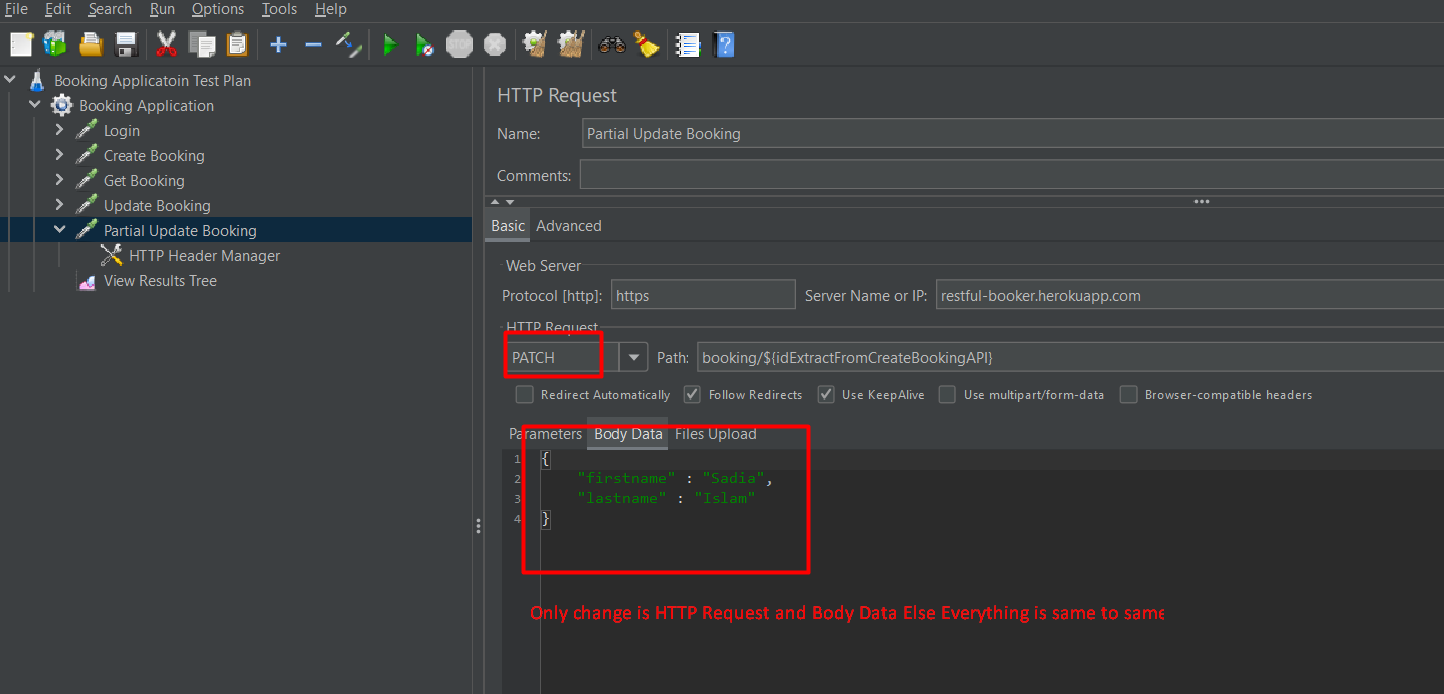
token = ${tokenExtractFromLogin} which is form Login JSON Extract file



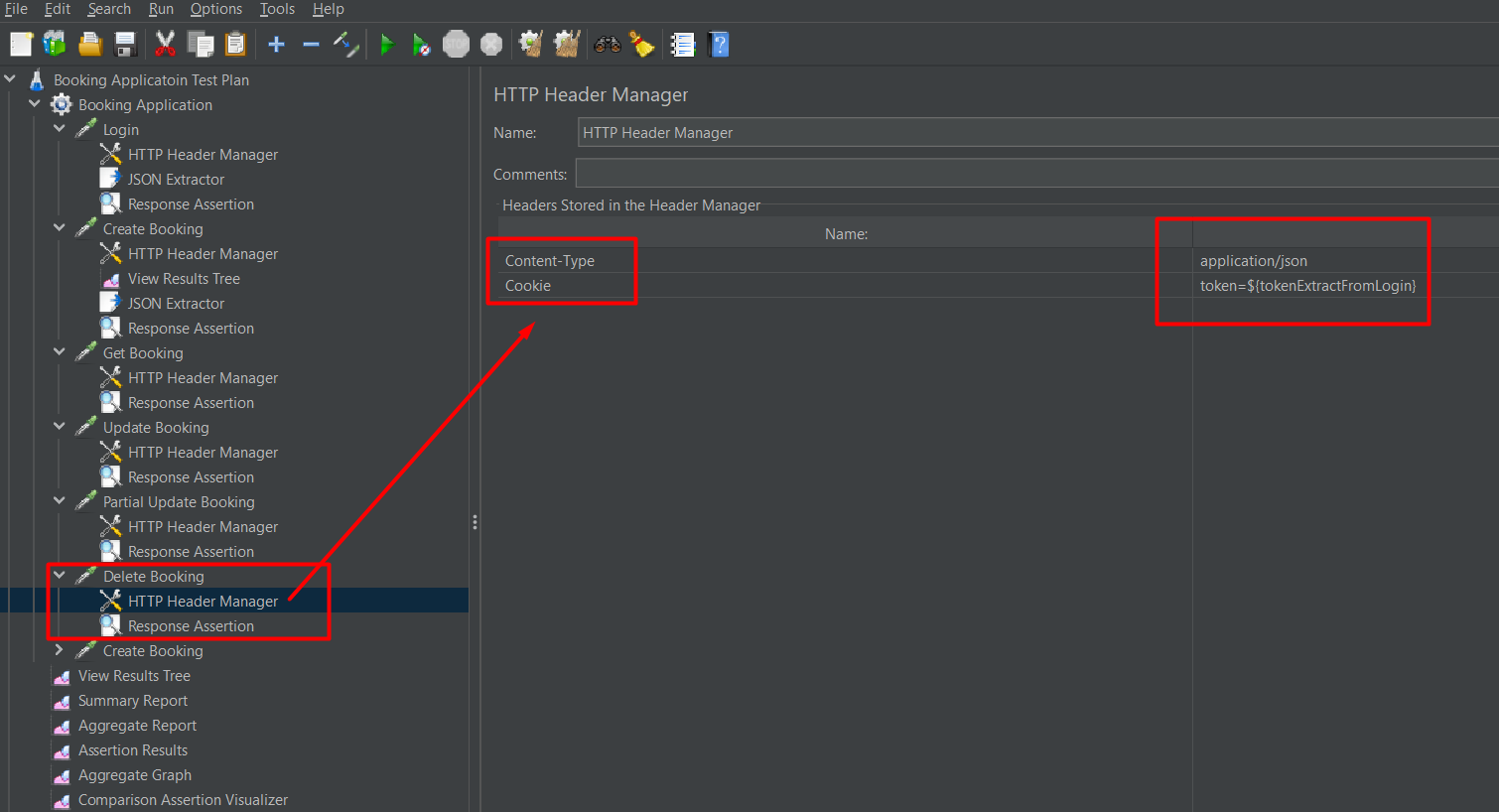
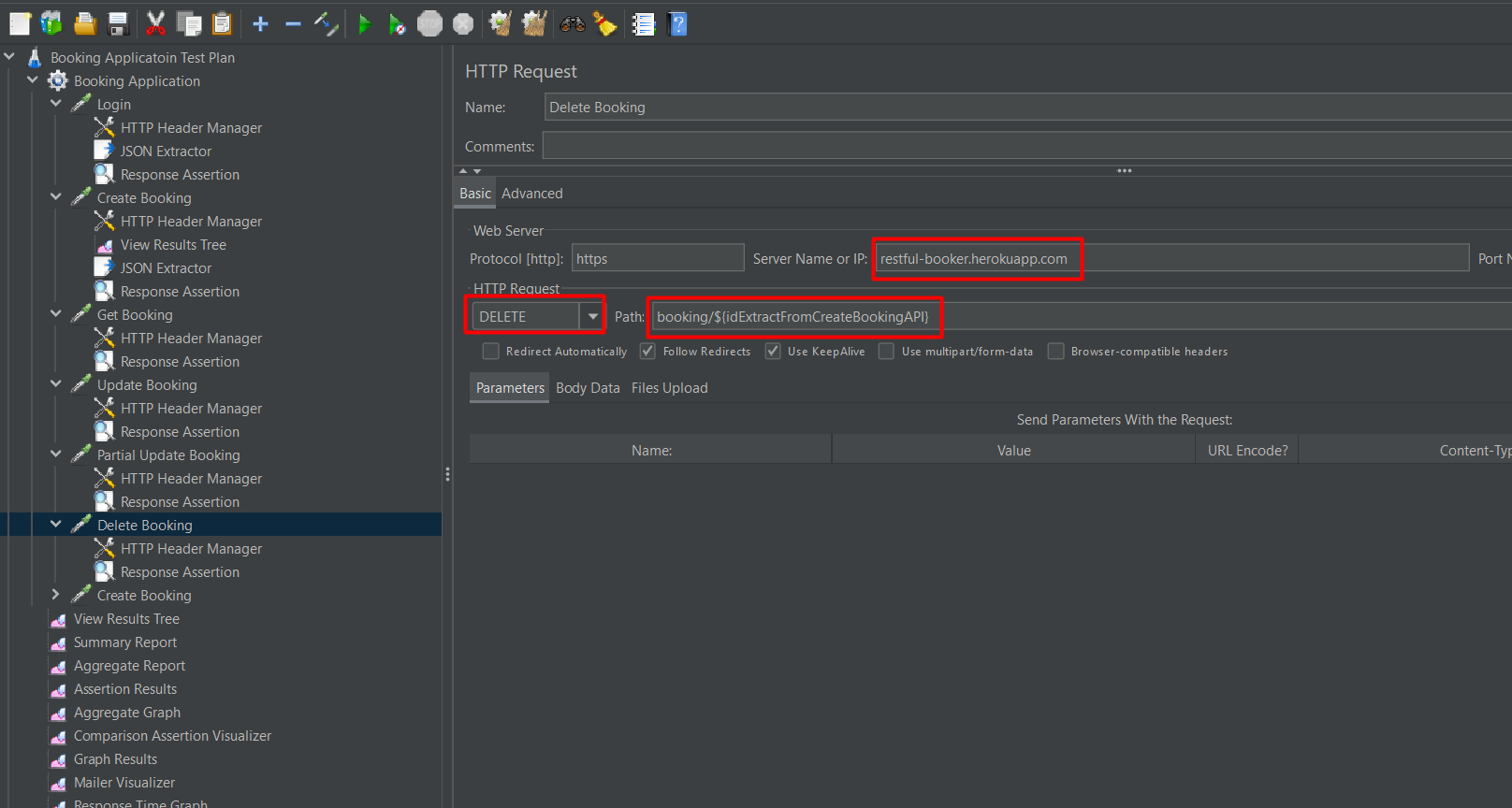


* 1. Doing for Partial Update Booking –
* **Given Documentation**:
* **API –** same as Update Booking
* **Method** – PATCH
* **Header** – follow update booking same to same nothing changes

Only change in body data-

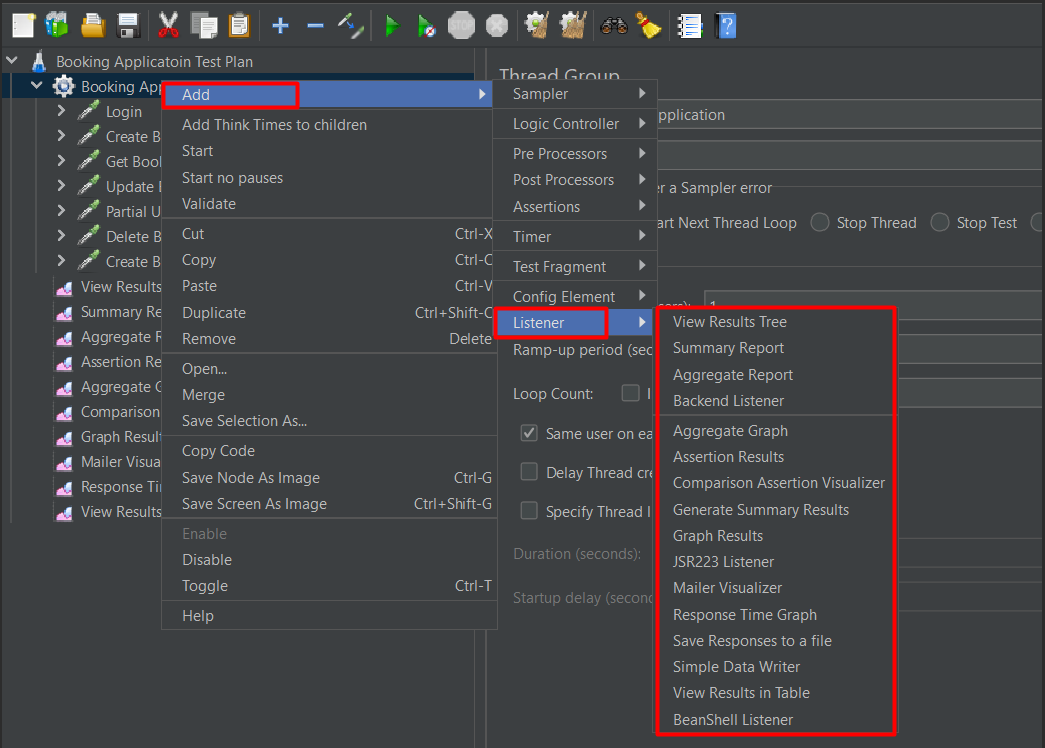


* 1. Doing for Delete Booking –
* **Given Documentation**:
* **API –** same as Update Booking
* **Method** – DELETE
* **Header** – follow partial update booking same to same nothing changes



1. **Listener Add** – component that shows the results of the samples**.**

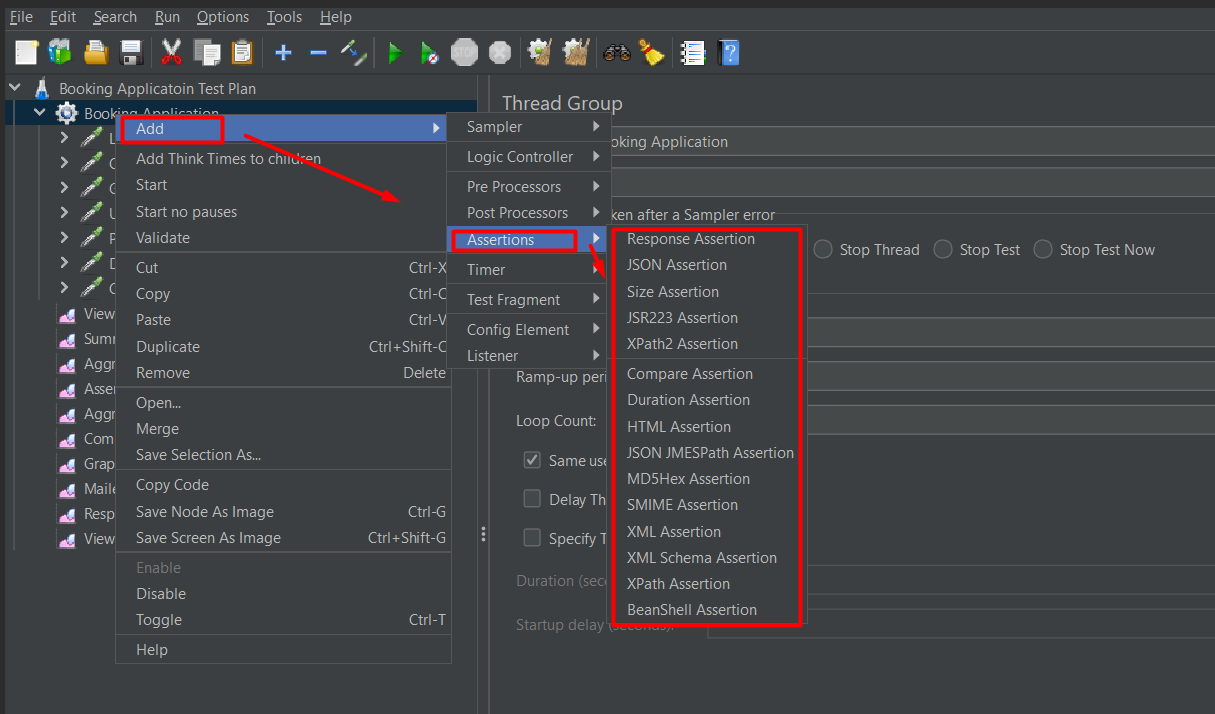
Some of the listeners are –

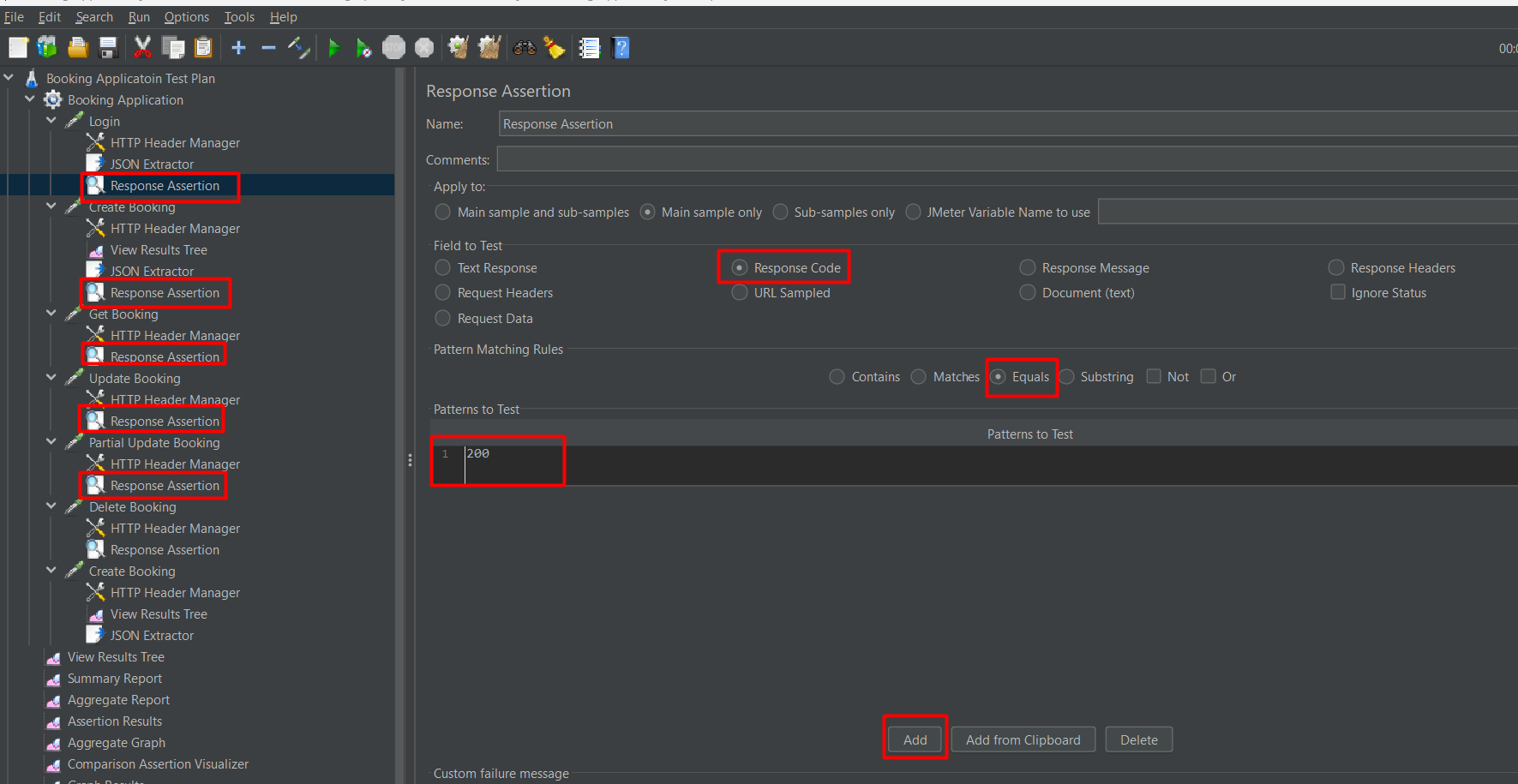


Its good to add listeners at the very end inside mother tree (means Booking Application) for getting final outputs. But in need we can use inside any sampler.

1. **Assertions Add** – used to validate the response to the request you sent to the server.

Some of the assertions are –



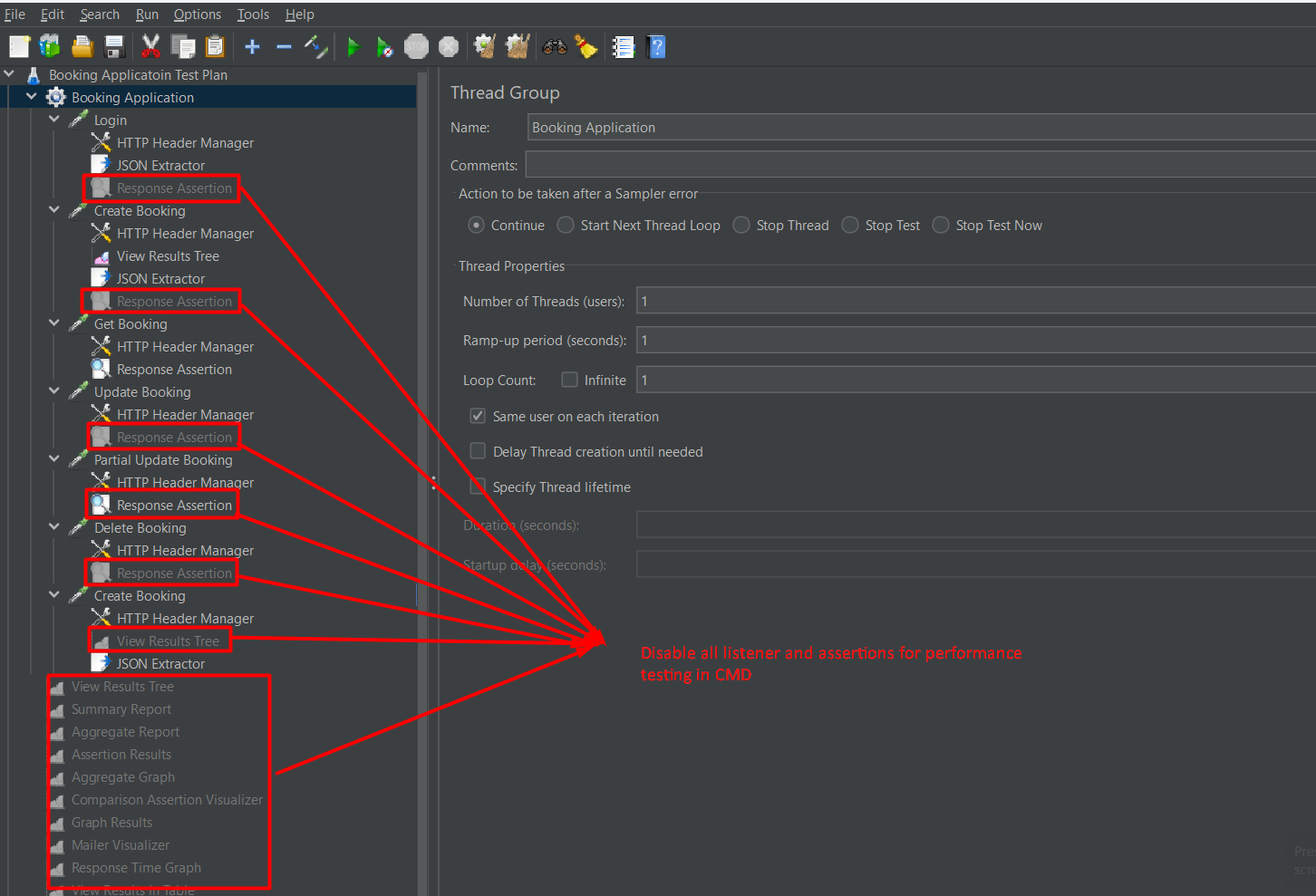


Use assertions as per documentations or required.

* In documentation from (**login to partial update booking**) response code = **200**.
* For **Delete booking** response code = **201** has been used.
* **Performance Testing using Command Line :**

**Step 1:** Save the jmx file with proper threads count/ ramp-up/ loop count

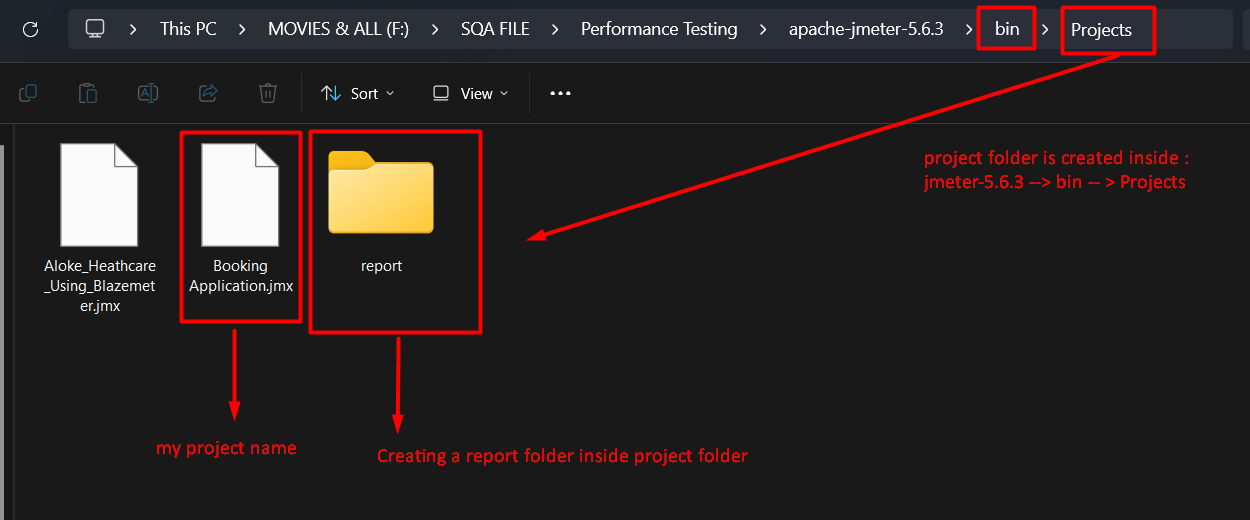
Keep in mind 🡪 all listener & assertions has to be disable before performance testing. Save it with a new name , we will do performance testing with that new saved file



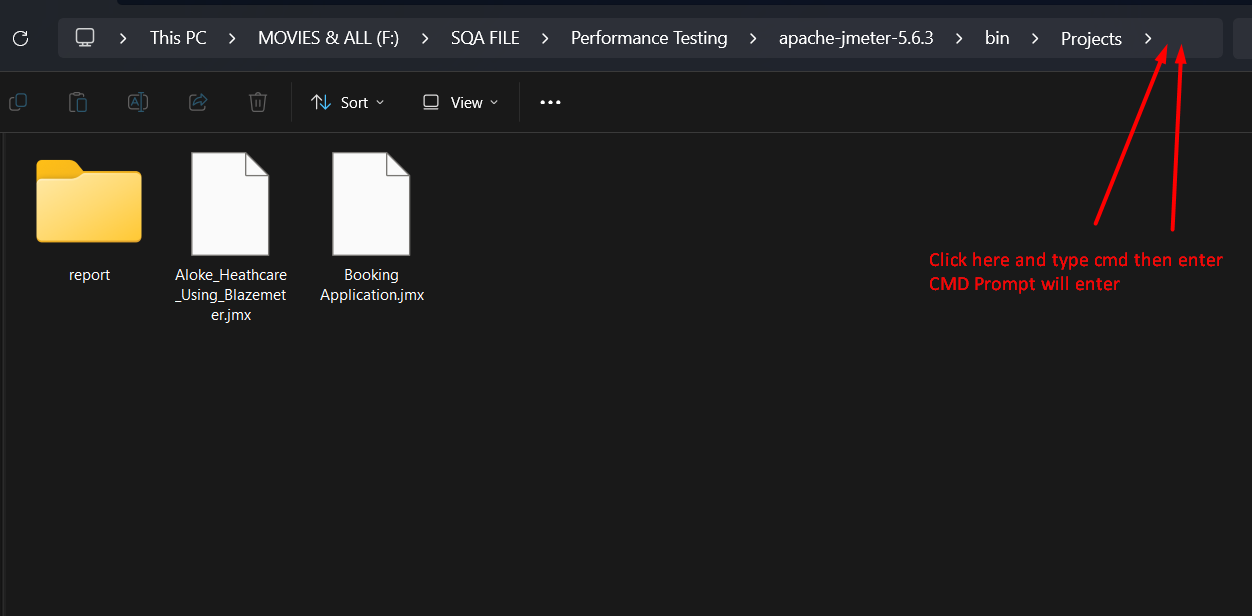
**Step 2:** Generate the command to run performance test in cmd –

Before that create a folder called report in bin folder or the folder where you save your jmx file which will store the report files with .**jtl** extension.

In my case I have created a folder (**name - project**) inside bin folder so I need to create the folder (**name - report**) inside project folder.



Now open cmd in this folder (**project folder**). First click on the address bar and type cmd and enter .



Now type below code and press enter.

jmeter -n -t BasicProject18\_t20.jmx -l report\BasicProject18\_t20.jtl

jmeter -g report\BasicProject18\_t20.jtl -o report\BasicProject18\_t20.html

1st Red marked BasicProject18\_t20.jmx represent project name and 2nd red marked BasicProject18\_t20 represent name for report file. Remember to rename the files name with your own file name. For my case it will be –

First Step:

jmeter -n -t BookingApplicationTesting.jmx -l report\BookingApplicationTesting.jtl

Second Step:

jmeter -g report\BookingApplicationTesting.jtl -o report\BookingApplicationTesting.html

Here,

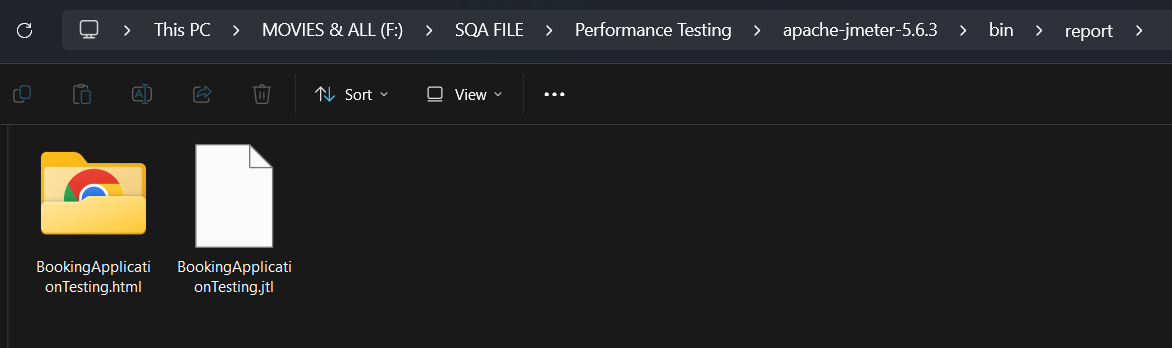
- n = non - GUI mode

- t = load the test plan located

- l = save the test results location

- g = generate

- o = output



Report file is ready go inside html folder and open index.html, you will see the full report