Mobiquity Java Assignment:

Please refer <https://github.com/Tanvi94/job_Assignments/blob/master/assignment_java-Mobiquity/skeleton_java/Packaging%20Challenge.pdf>

for problem statement.

There are 2 API’s created:

1. Set the Weight Limit and take that weight limit from user.
2. Calculate index numbers of possible packages which will get included in resulting gift package which will be equal to or less than weight limit and that will have maximum cost.

API details:

* Set the Weight Limit and take that weight limit from user.

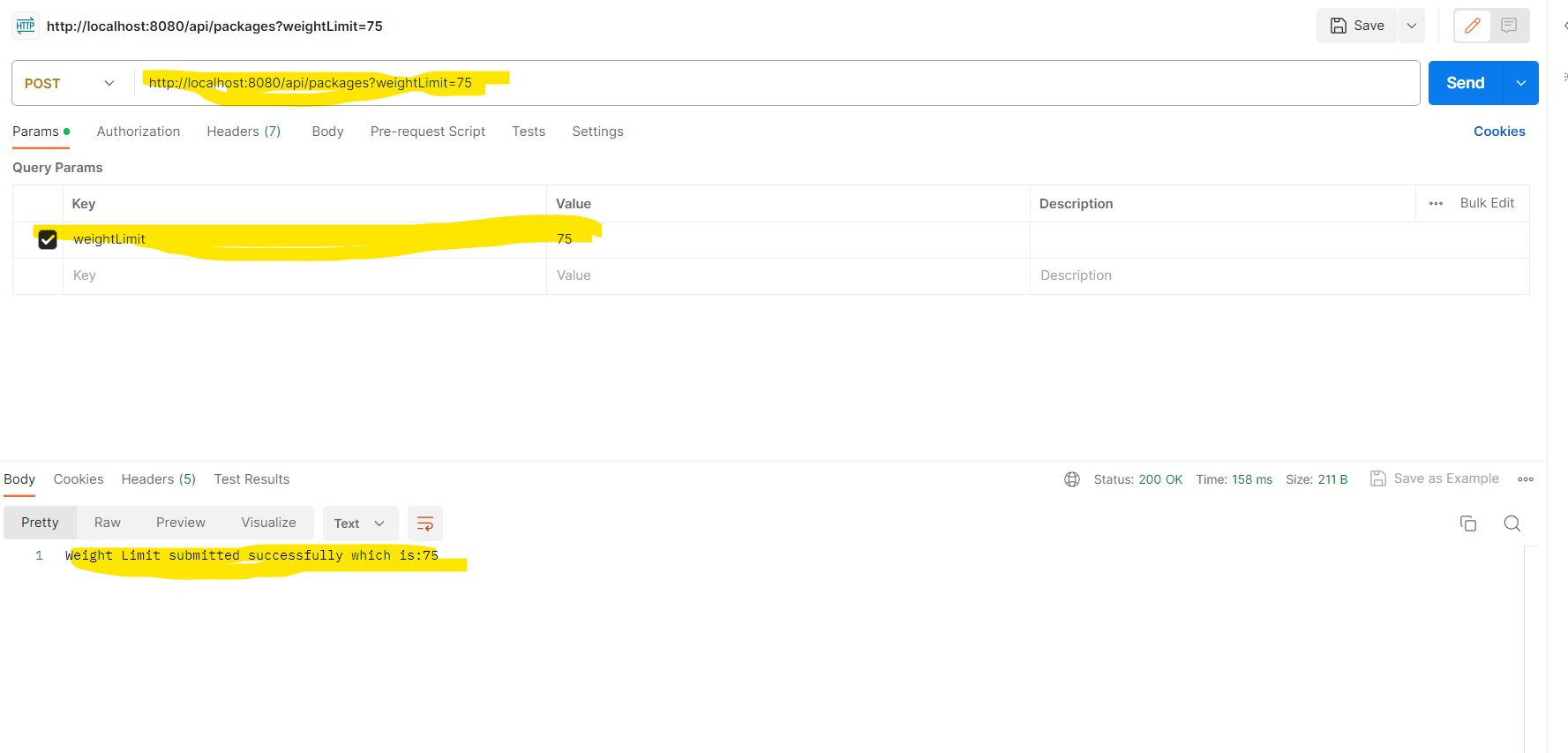
Mapping: /api/packages

Method: POST

Input: weightLimit to set for the gift package as a request parameter.

Output: returns weightLimit value (in String).

API call: <http://localhost:8080/api/packages>

Execution/Testing through Postman: 

* Calculate index numbers of possible packages which will get included in resulting gift package which will be equal to or less than weight limit and that will have maximum cost.

As it was a static API, I have created REST API and uploaded file through this API as a input which contains data of various packages.

Mapping: /api/call-static-api

Method: POST

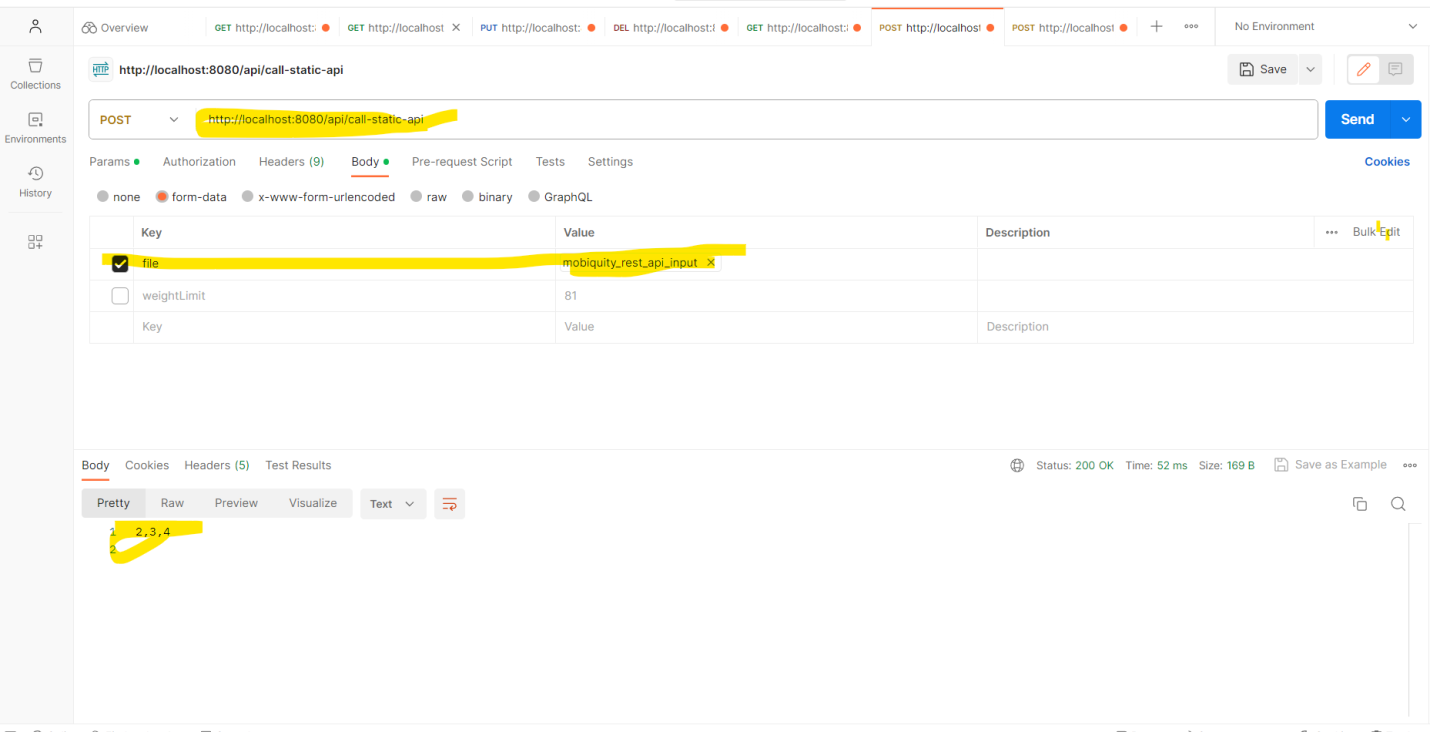
Input: file which contains package information

Output: Returns indexes of packages included in the Gift package in String format

API call: <http://localhost:8080/api/call-static-api>

While providing input as a file, Please use “form-data” option from request body and in Postman give file as a key and select “File” option and in value section select input file to upload.

Execution/Testing through Postman:



Strategy/Data Structure Used in the application:

This API’s are designed in Spring Boot Application as it provides ease and flexibility to design API’s as well as there are many annotations which are helping to design API’s easily.

We have to provide packages such that the weight of that all packages should be less than or equal to total weight provided through API and total cost is as large as possible.

It is similar to 0/1 knapsack problem of dynamic programming.

That’s why, same logic is applied to get index numbers of all packages that fulfills given conditions.

* Limitations:

1. Most of the validations are handled but there is a scope of improvement.

* Assumptions:

1. File format is fixed.

* Improvements:

1. API’s Security and Unit tests cases should be considered at the time of deployment.