



2020-2021 Semester 2

CT545 Enterprise Java Programming

Assignment 5

Screenshots of API testing in Postman and Browser-based Client

**Jiarong Li
J.Li11@nuigalway.ie
20230033
1MF1**

Content

Part 1: Screenshots of API testing in Postman ----- 3

1. GET request, return the list of details of all bookings in the system in JSON format. ----- 3
2. GET request, return details of the booking with the specified id in JSON format. ----- 4
3. POST request, create a new instance of Booking, and add it to the list of Bookings., and return details of the Booking that was just added in JSON format. ----- 5,6
4. PUT request, update the details of the booking with the specified id, and return details of the Booking that was just updated in JSON format. ----- 7,8
5. DELETE request, delete the Booking with the specified id ----- 9

Part 2: Screenshots of API testing in Brower-based Client ----- 10

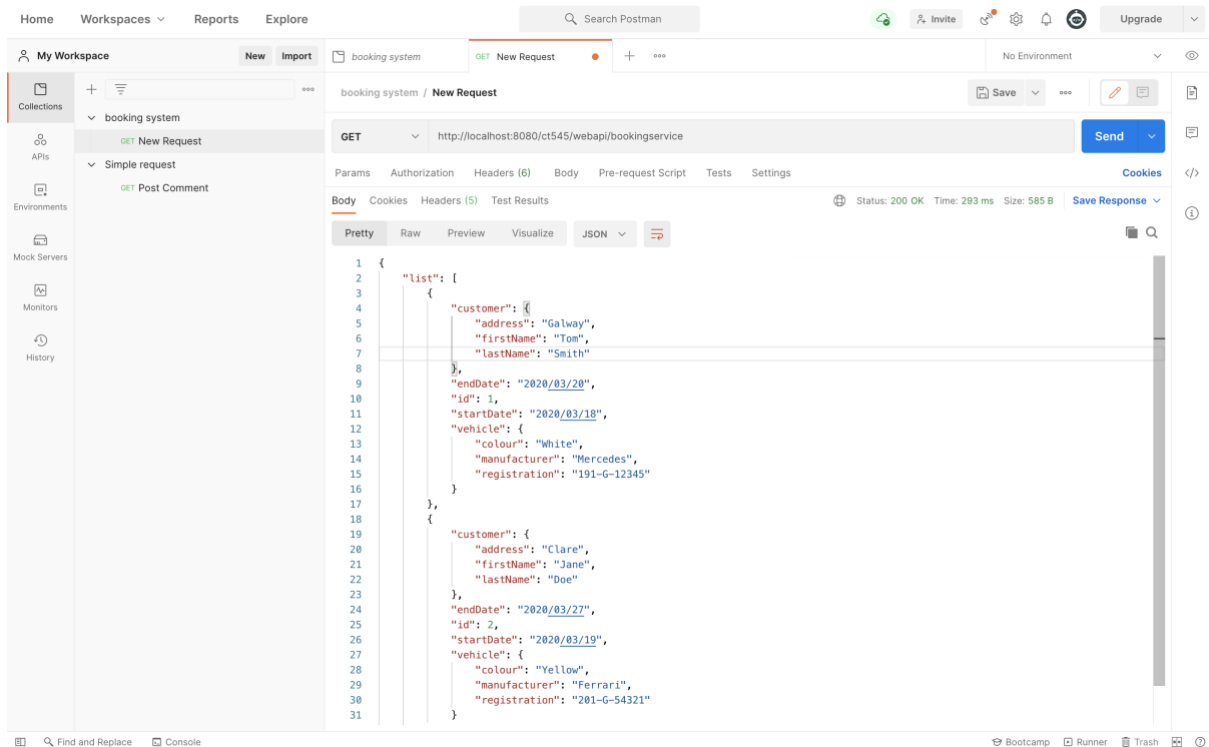
1. The main menu. ----- 10
2. View all bookings. ----- 10
3. View a single booking. ----- 10,11
4. create a new booking. ----- 11,12
5. Delete an existing booking. ----- 12,13
6. Update an existing booking. ----- 13,14

Part 1: Screenshots of API testing in Postman

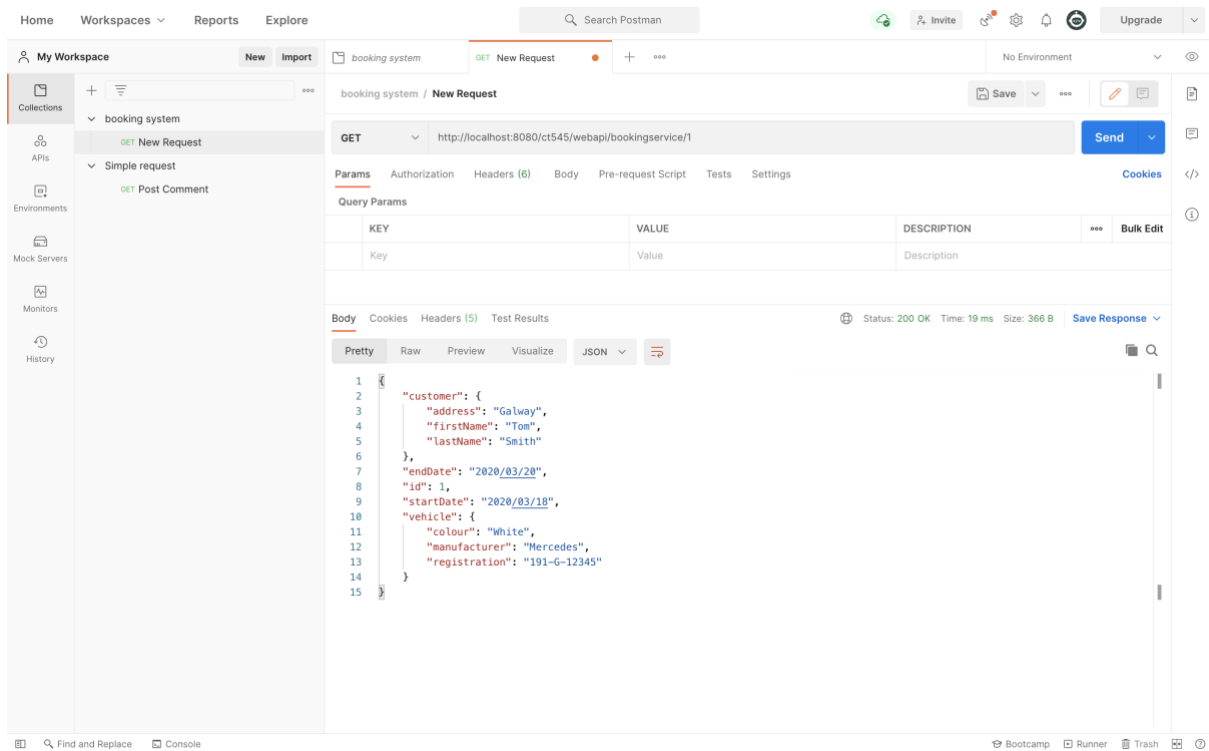
Use the Postman app to test the functionality of each of items 1 – 5 below. Include screenshots showing the results of our tests along with our code.

1. When a GET request is received at the URL

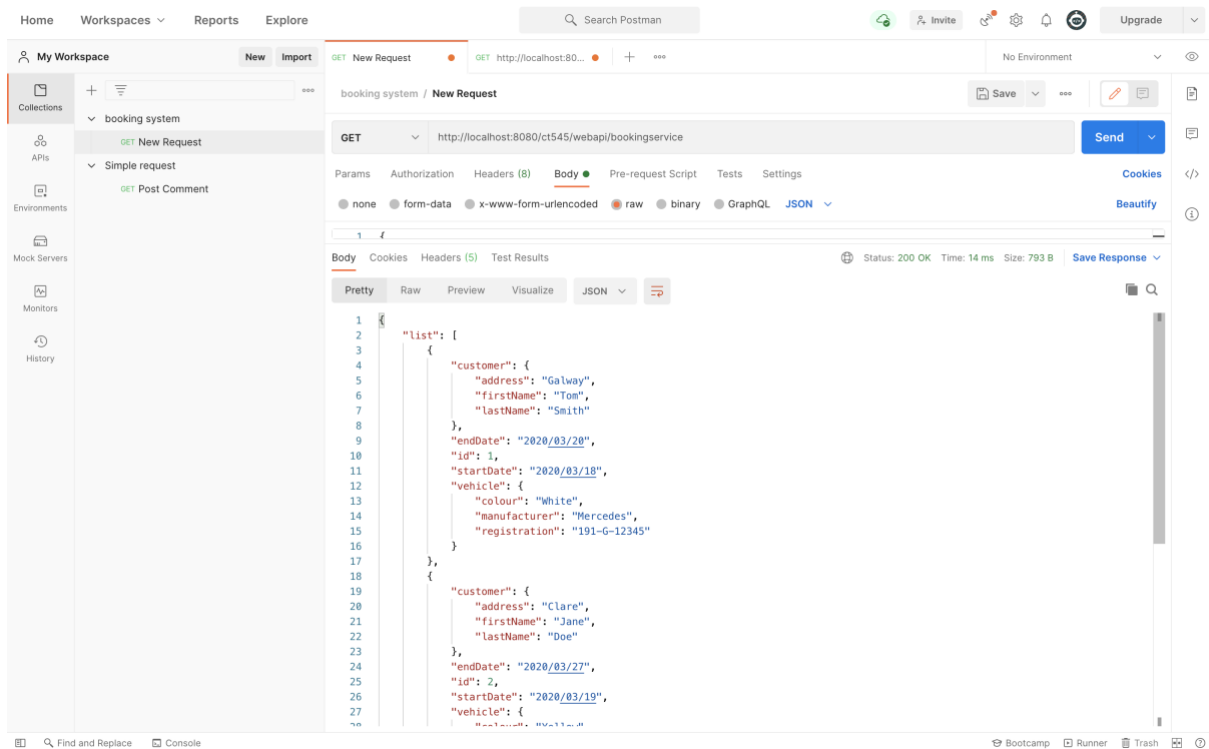
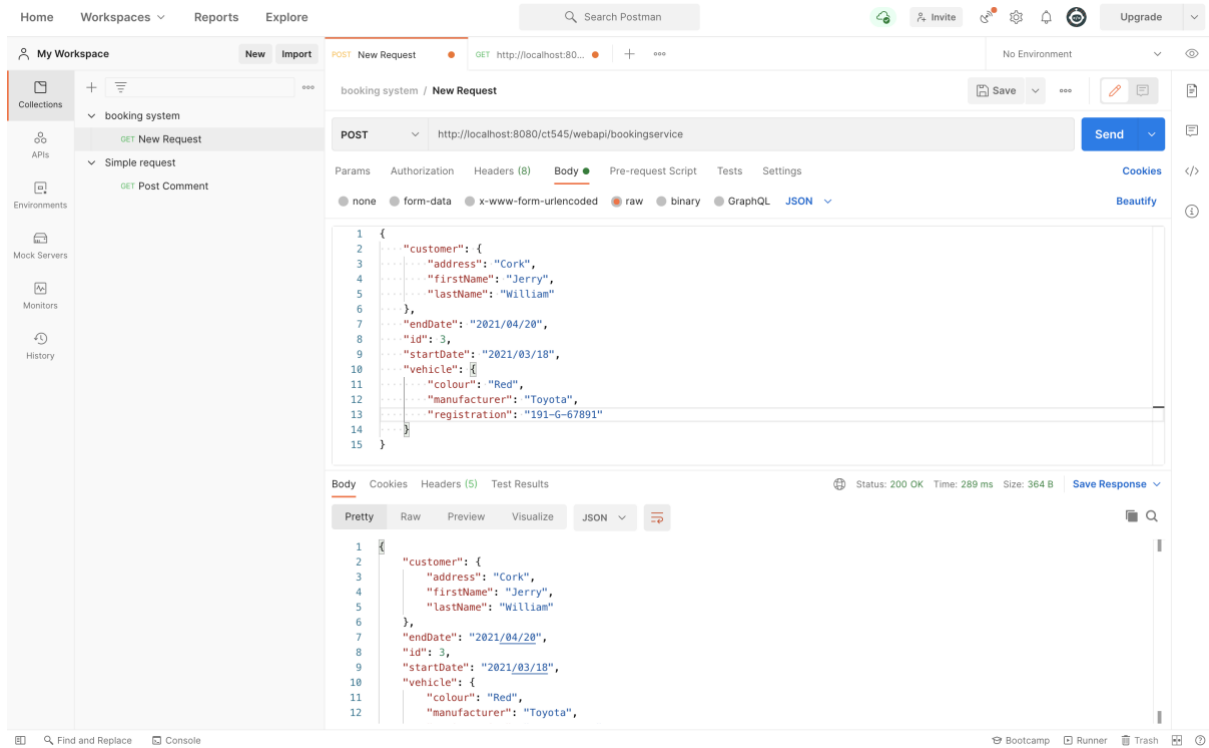
<http://localhost:8080/ct545rest/webapi/bookingservice> the service should return the list of details of all bookings in the system in JSON format.



2. When a GET request is received at the URL <http://localhost:8080/ct545rest/webapi/bookingservice/{id}> the service should return details of the booking with the specified id in JSON format.



3. When a POST request is received at the URL <http://localhost:8080/ct545rest/webapi/bookingservice> the service should create a new instance of Booking a JSON representation from the body of a POST request, and add it to the list of Bookings. This method should return details of the Booking that was just added in JSON format.



Home Workspaces Reports Explore

Search Postman

My Workspace New Import

Collections

- booking system
 - GET New Request
 - Simple request
 - GET Post Comment

APIs

Environments

Mock Servers

Monitors

History

GET New Request

http://localhost:8080/ct545/webapi/bookingservice

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 14 ms Size: 793 B Save Response

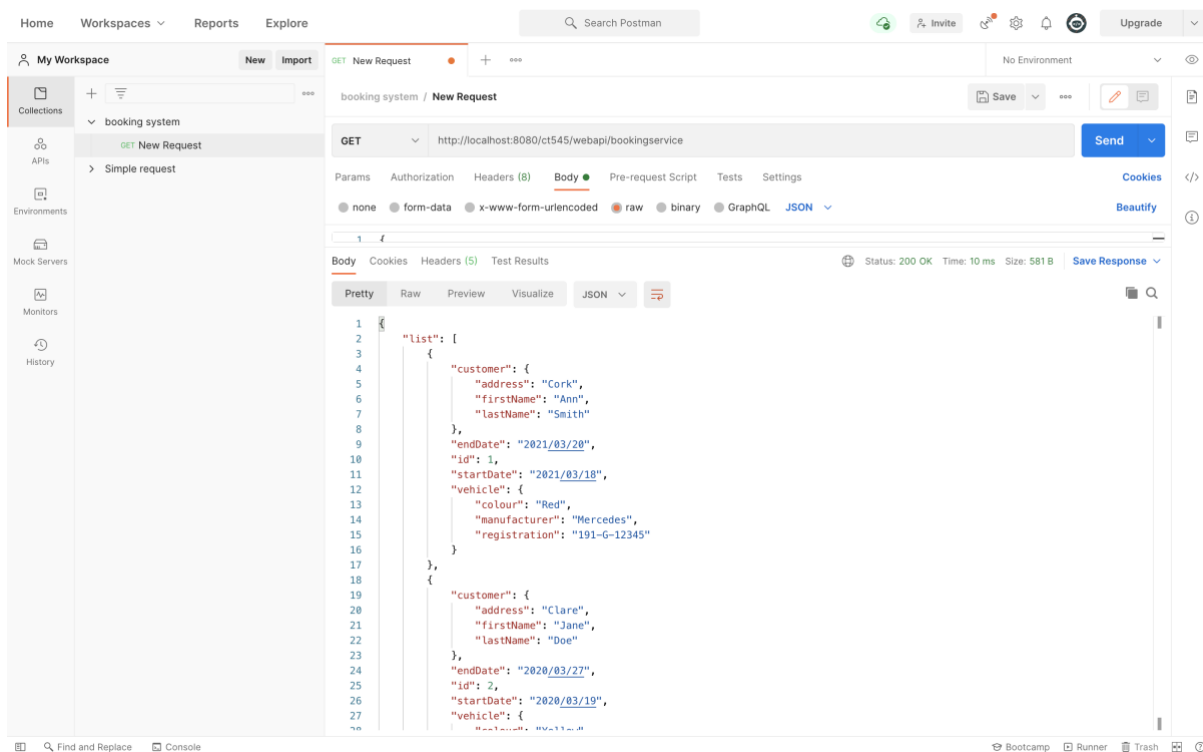
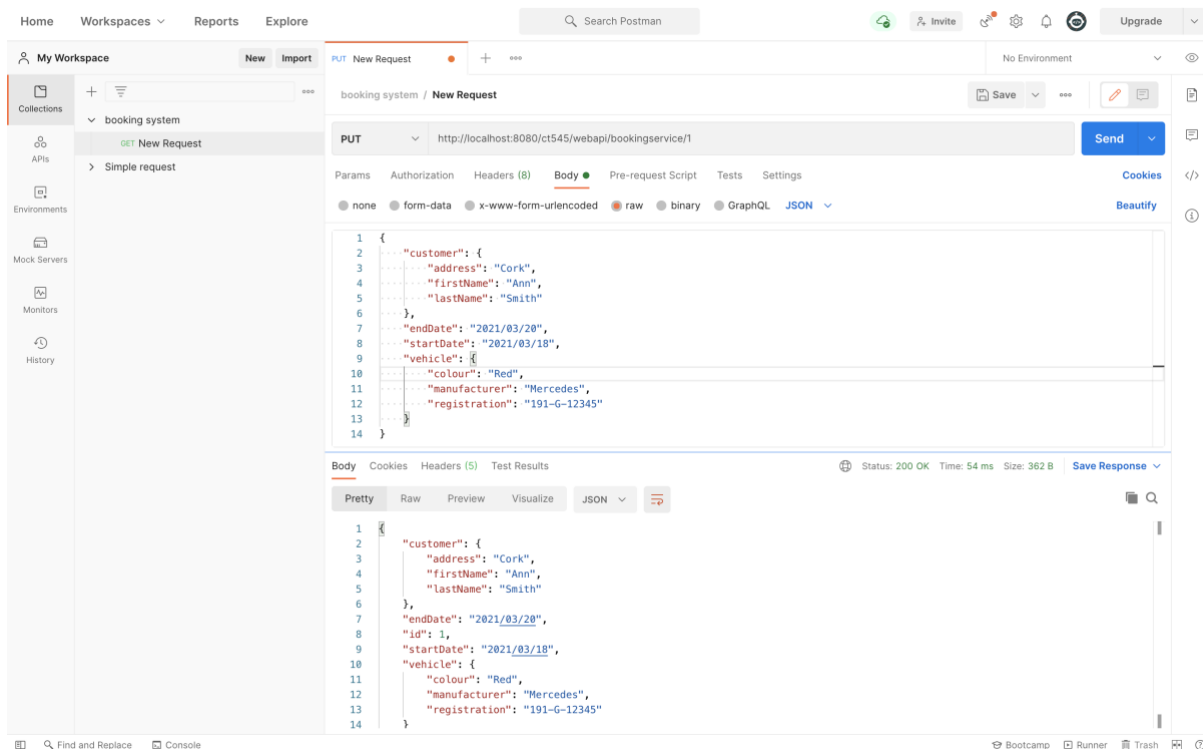
Pretty Raw Preview Visualize JSON

```
23      },
24      "endDate": "2020/03/27",
25      "id": 2,
26      "startDate": "2020/03/19",
27      "vehicle": {
28        "colour": "Yellow",
29        "manufacturer": "Ferrari",
30        "registration": "201-G-54321"
31      }
32    },
33    {
34      "customer": {
35        "address": "Cork",
36        "firstName": "Jerry",
37        "lastName": "William"
38      },
39      "endDate": "2021/04/20",
40      "id": 3,
41      "startDate": "2021/03/18",
42      "vehicle": {
43        "colour": "Red",
44        "manufacturer": "Toyota",
45        "registration": "191-G-67891"
46      }
47    }
48  ]
49 }
```

Find and Replace Console

Bootcamp Runner Trash

4. When a PUT request is received at the URL <http://localhost:8080/ct545rest/webapi/bookingservice/{id}> the service should update the details of the booking with the specified id using the JSON representation in the body of the PUT request. This method should return details of the Booking that was just updated in JSON format.



Home Workspaces Reports Explore Search Postman

My Workspace New Import

collections + booking system

GET New Request

Simple request

Environments Mock Servers Monitors History

booking system / New Request

GET http://localhost:8080/ct545/webapi/bookingservice

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1

Body Cookies Headers (5) Test Results Status: 200 OK Time: 10 ms Size: 581 B Save Response

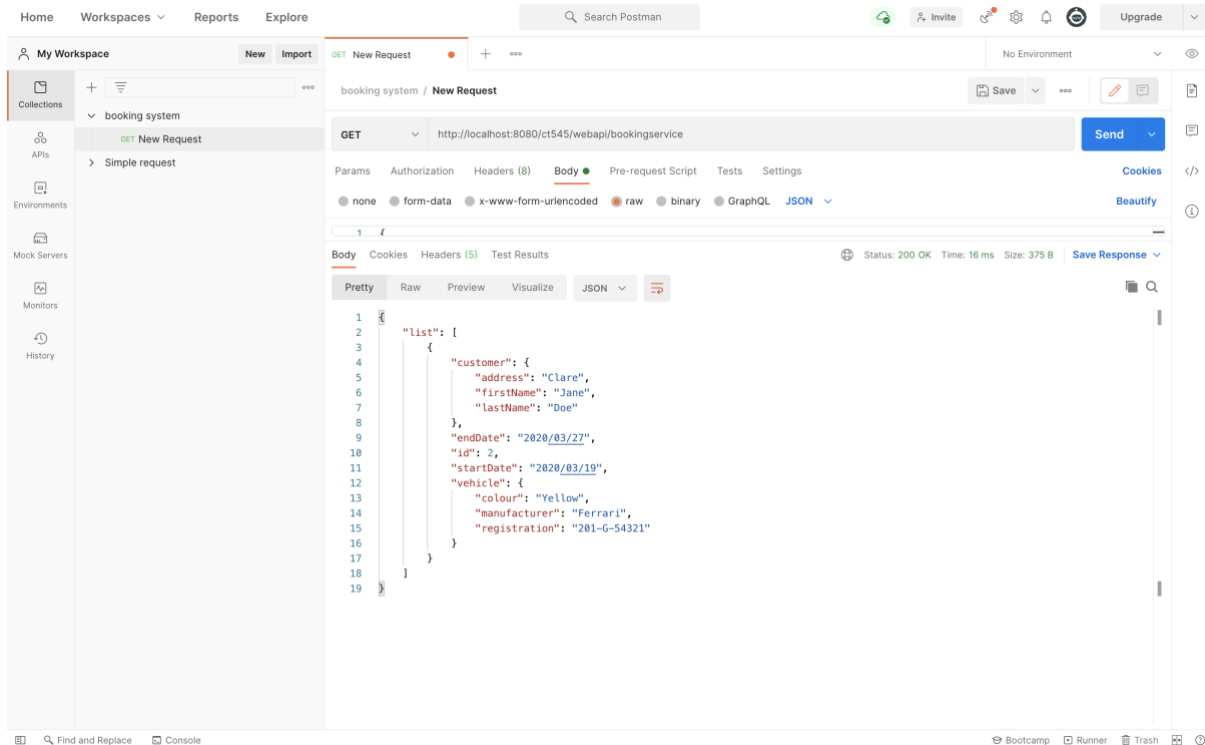
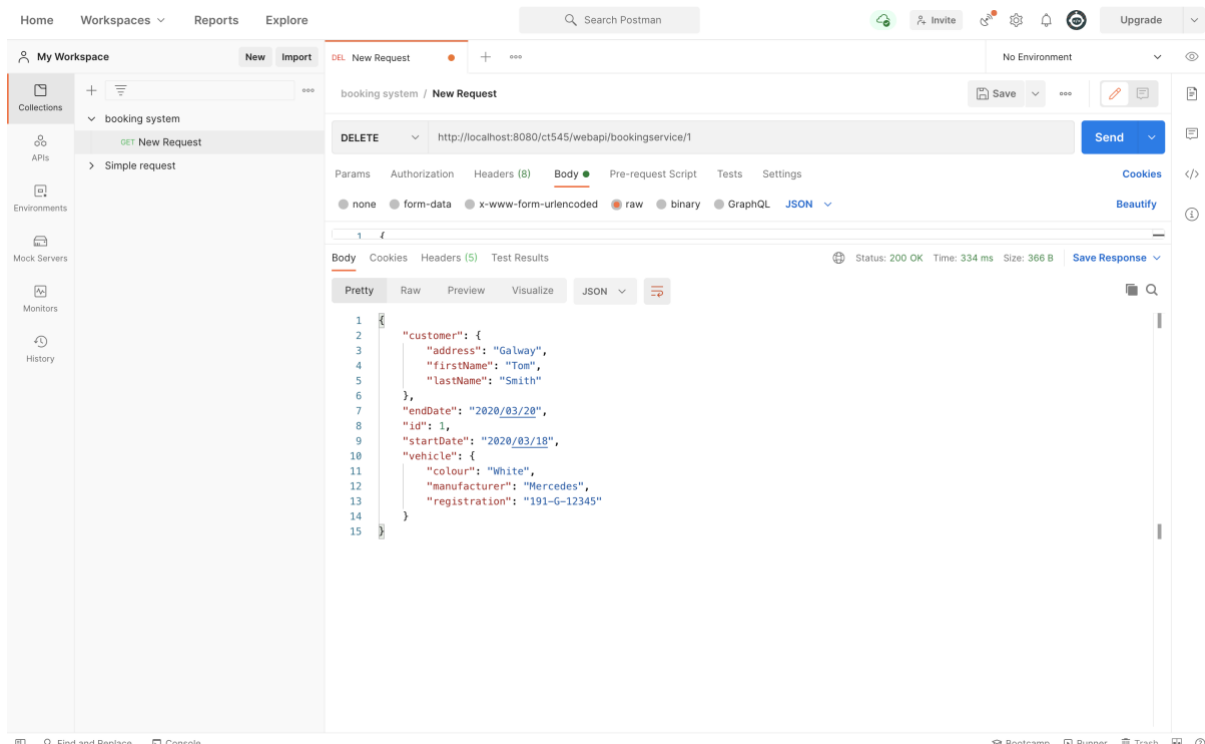
Pretty Raw Preview Visualize JSON

```
8      },
9      "endDate": "2021/03/20",
10     "id": 1,
11     "startDate": "2021/03/18",
12     "vehicle": {
13       "colour": "Red",
14       "manufacturer": "Mercedes",
15       "registration": "191-G-12345"
16     }
17   },
18   {
19     "customer": {
20       "address": "Clare",
21       "firstName": "Jane",
22       "lastName": "Doe"
23     },
24     "endDate": "2020/03/27",
25     "id": 2,
26     "startDate": "2020/03/19",
27     "vehicle": {
28       "colour": "Yellow",
29       "manufacturer": "Ferrari",
30       "registration": "201-G-54321"
31     }
32   }
33 }
34 }
```

End and Barrels Console

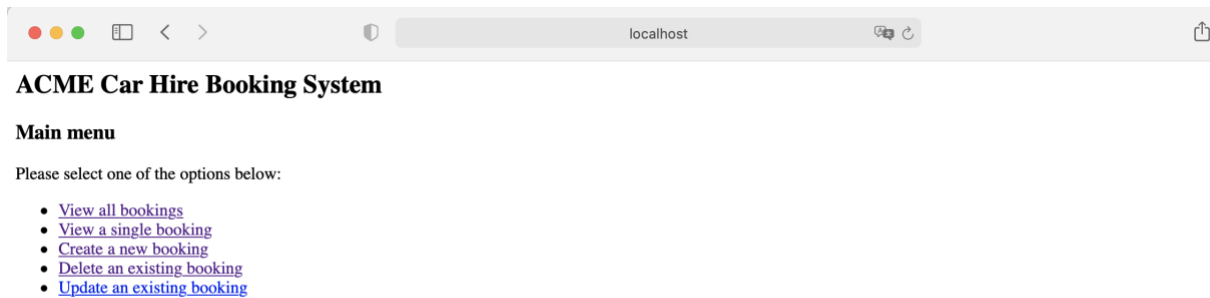
Entrance Docker Trash

5. When a DELETE request is received at the URL <http://localhost:8080/ct545rest/webapi/bookingservice/{id}> the service should delete the Booking with the specified id

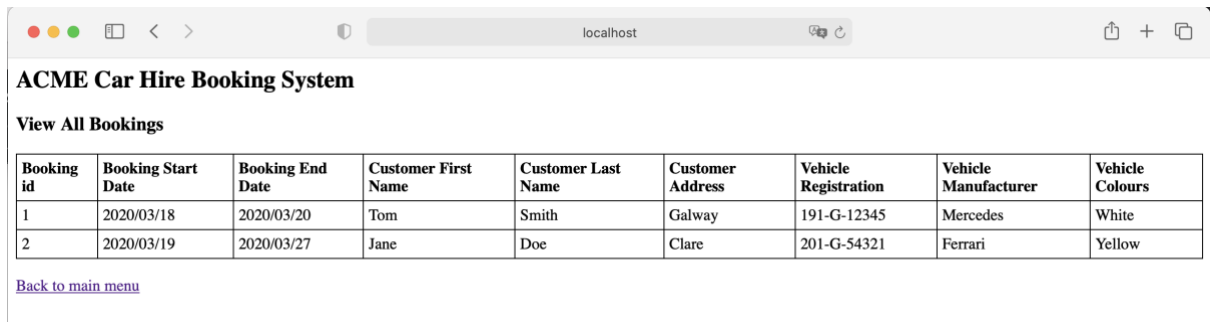


Part 2: Screenshots of API testing in Brower-based Client

1. The main menu.

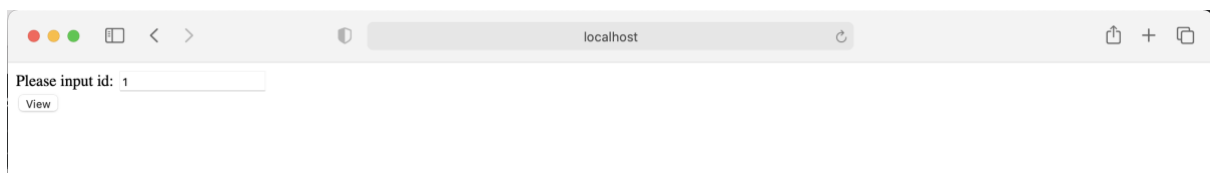


2. View all bookings

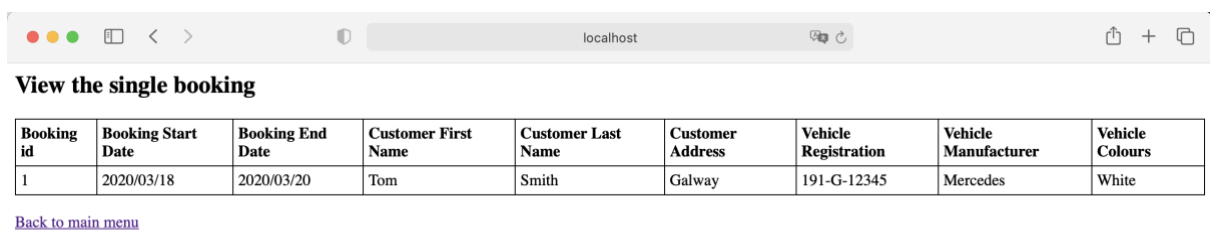


3. View a single booking

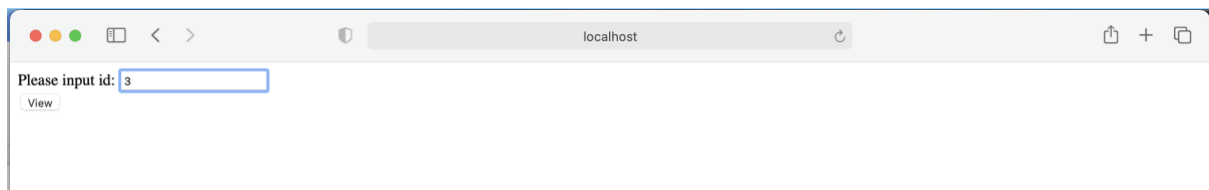
3.1 Input the booking id



3.2 Click the submit button

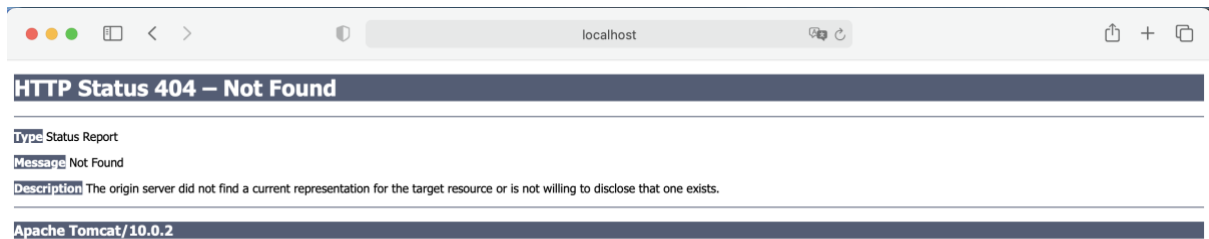


3.3 If the booking id does not exist



A web browser window with the address bar set to 'localhost'. The page contains a form with the text 'Please input id:' followed by a text input field containing the number '3'. Below the input field is a button labeled 'View'.

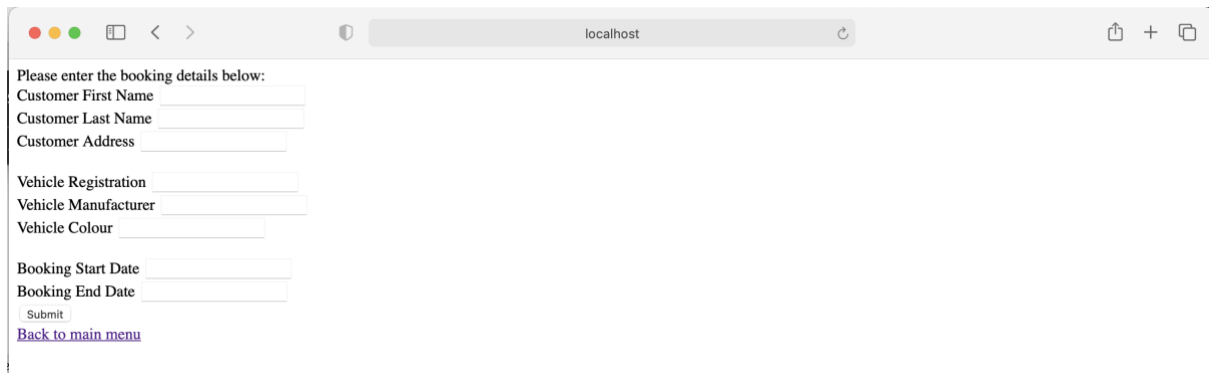
When we click the submit button



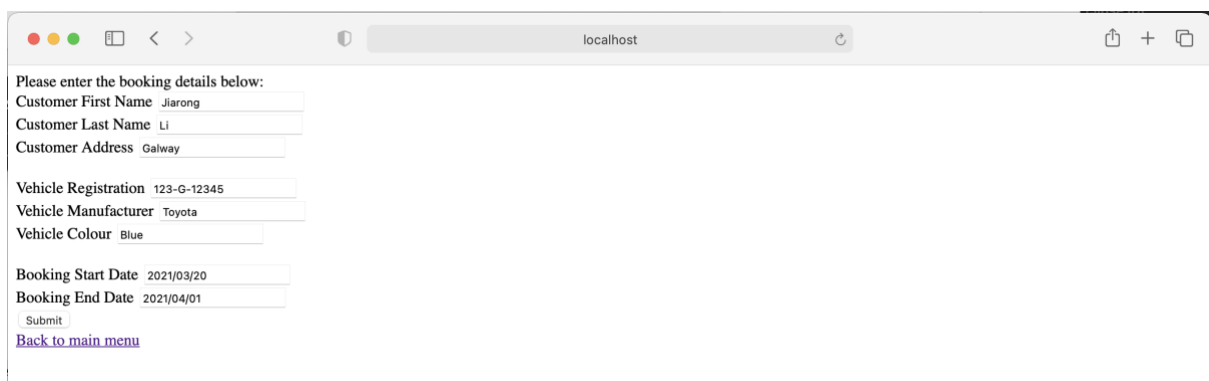
A web browser window showing an HTTP 404 - Not Found error. The page has a dark blue header with the text 'HTTP Status 404 – Not Found'. Below the header, the page content includes: 'Type: Status Report', 'Message: Not Found', and 'Description: The origin server did not find a current representation for the target resource or is not willing to disclose that one exists.' At the bottom, it says 'Apache Tomcat/10.0.2'.

4. Create a new booking

4.1 Input the information

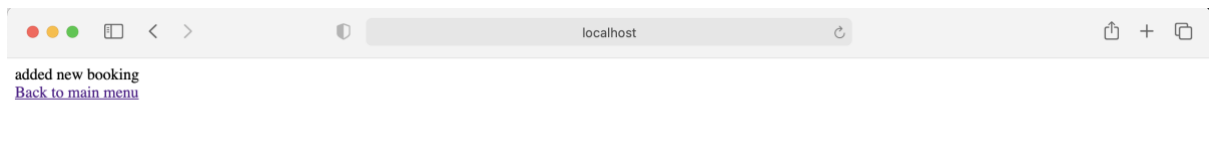


A web browser window showing a form to input booking details. The form has the following fields: 'Customer First Name', 'Customer Last Name', 'Customer Address', 'Vehicle Registration', 'Vehicle Manufacturer', 'Vehicle Colour', 'Booking Start Date', and 'Booking End Date'. Each field has a corresponding text input box. Below the 'Booking End Date' field is a 'Submit' button and a link labeled 'Back to main menu'.

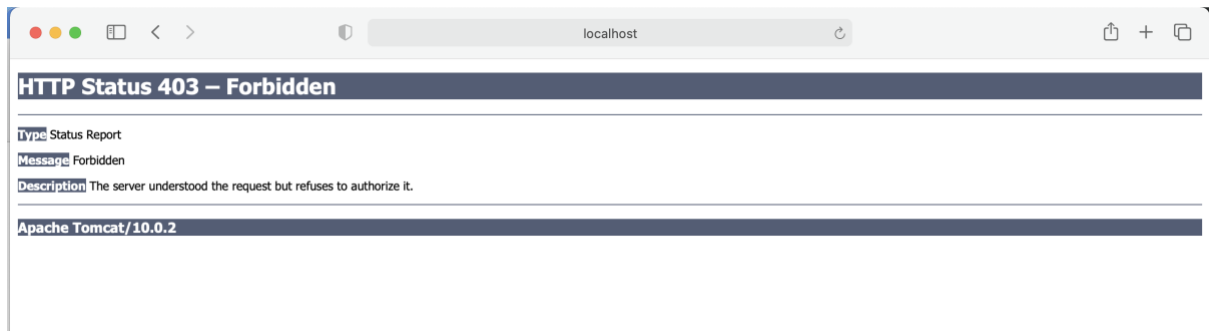


A web browser window showing the same form as above, but with input data. The 'Customer First Name' field contains 'Jiarong', 'Customer Last Name' contains 'Li', 'Customer Address' contains 'Galway', 'Vehicle Registration' contains '123-G-12345', 'Vehicle Manufacturer' contains 'Toyota', 'Vehicle Colour' contains 'Blue', 'Booking Start Date' contains '2021/03/20', and 'Booking End Date' contains '2021/04/01'. The 'Submit' button and 'Back to main menu' link are still present.

4.2 Clicked the submit button

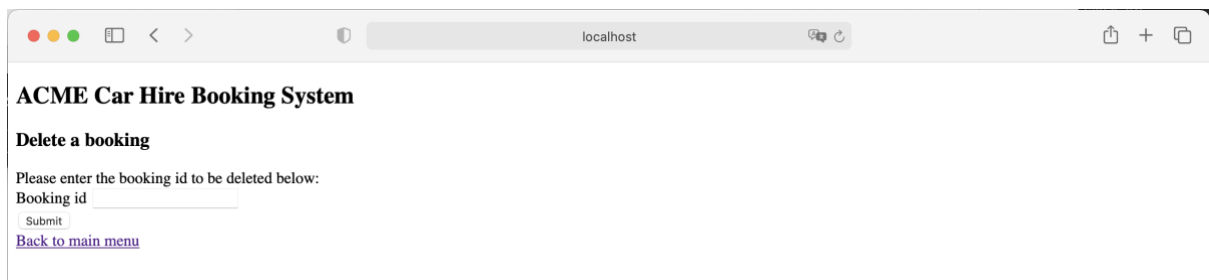


4.3 If there is the same booking in the list

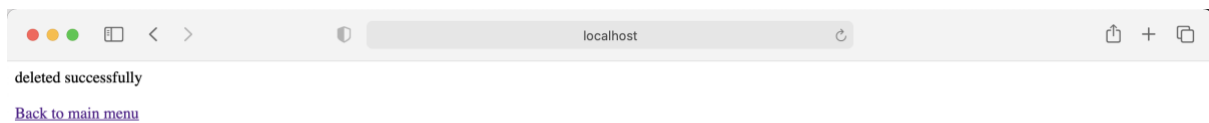


5. Delete an existing booking

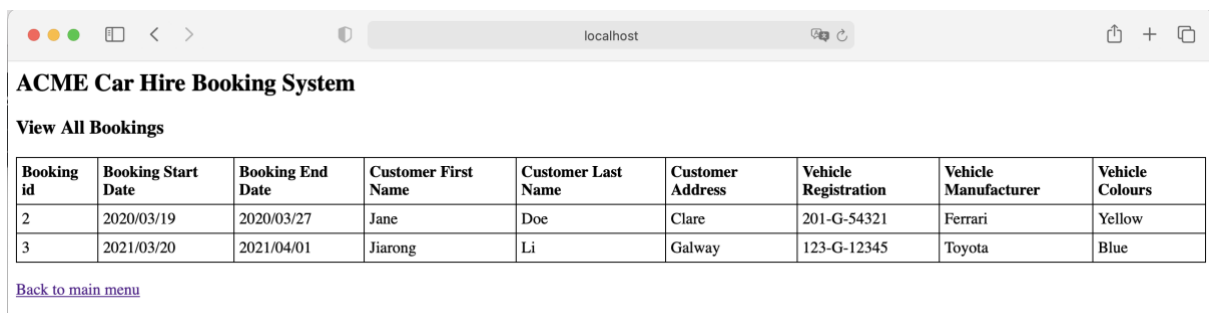
5.1 Input the booking id



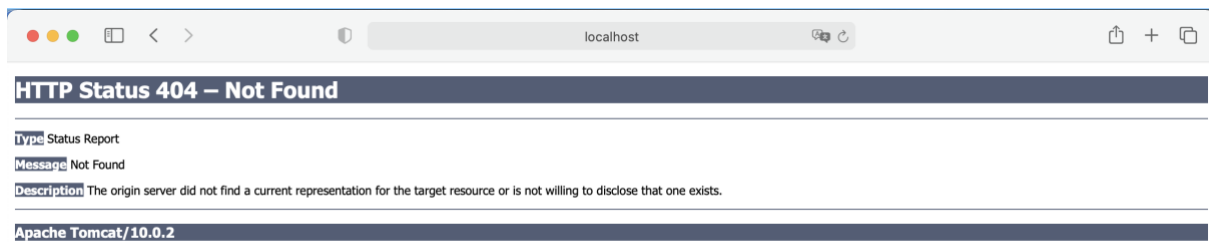
5.2 Click the submit button



5.3 Let's view all bookings now



5.4 If the booking id does not exist

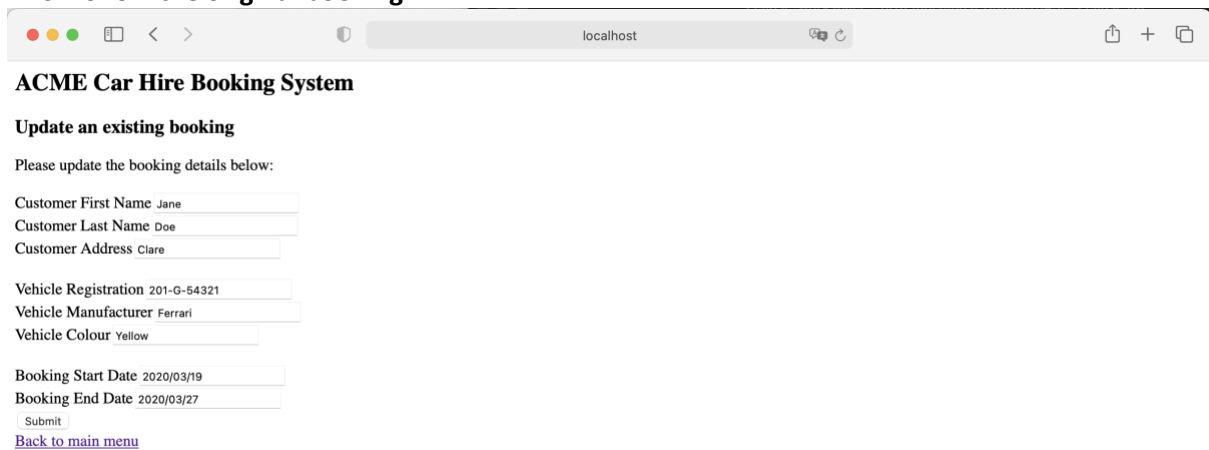


6. Update an existing booking

6.1 Input the booking id



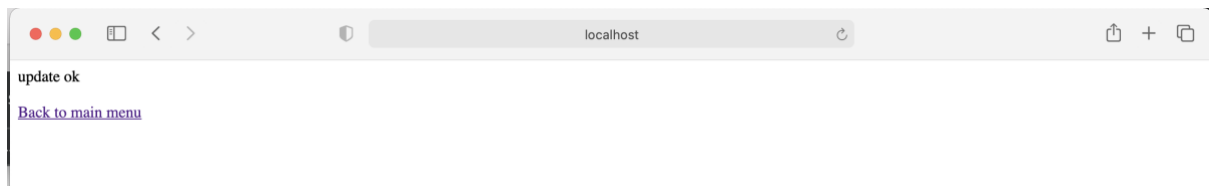
6.2 Show the original booking



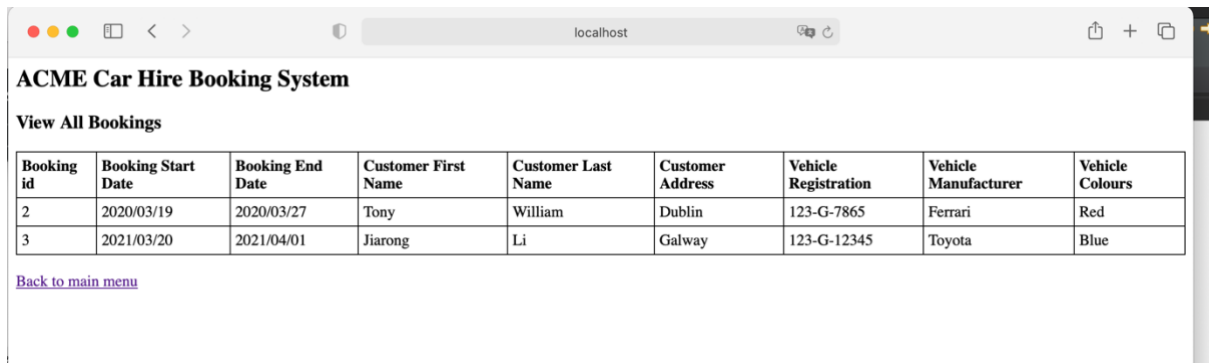
6.3 Edit the information in this booking



6.4 Clicked the submit button



6.5 Now we open the page of “View all bookings”, we can see the updated booking with id 2.



6.6 If the booking id does not exist

