WEB TECHNOLOGIES – WRITE UP
SPRINT #O

CONTENT

1. INTRODUCTION

The product category chosen to the project is Online Courses (LearnOn). The product has the following attributes: id, title, duration in hours, price, instructor, subject, description, type, level and image. This information will be stored in the database. The same database will have a table to store the users that will manage the courses. This table has the following attributes: username, password, name and image. To the sprint#0 is being delivered an implementation of RESTful that allow the CRUD operations to users and courses. All the data is being returned in JSON format and the postman rest client is being used to test the API.

2. USER STORIES

1. As an API Consumer, I want to add a course, so that I can add a new course through my client application.
2. As an API Consumer, I want to retrieve data in JSON format for all courses, so that I can use the data in my client application.
3. As an API Consumer, I want to retrieve data in JSON format for one course based on its ID, so that I can use the data in my client application.
4. As an API Consumer, I want to update a course, so that I can update an entry through my client application.
5. As an API Consumer, I want to delete a course, so that I can remove unwanted entries through my client application.
6. As an API Consumer, I want to add a user, so that I can add a new user through my client application.
7. As an API Consumer, I want to retrieve data in JSON format for all users, so that I can use the data in my client application.
8. As an API Consumer, I want to retrieve data in JSON format for one user based on its username, so that I can use the data in my client application.
9. As an API Consumer, I want to update a user, so that I can update an entry through my client application.
10. As an API Consumer, I want to delete a user, so that I can remove unwanted entries through my client application.

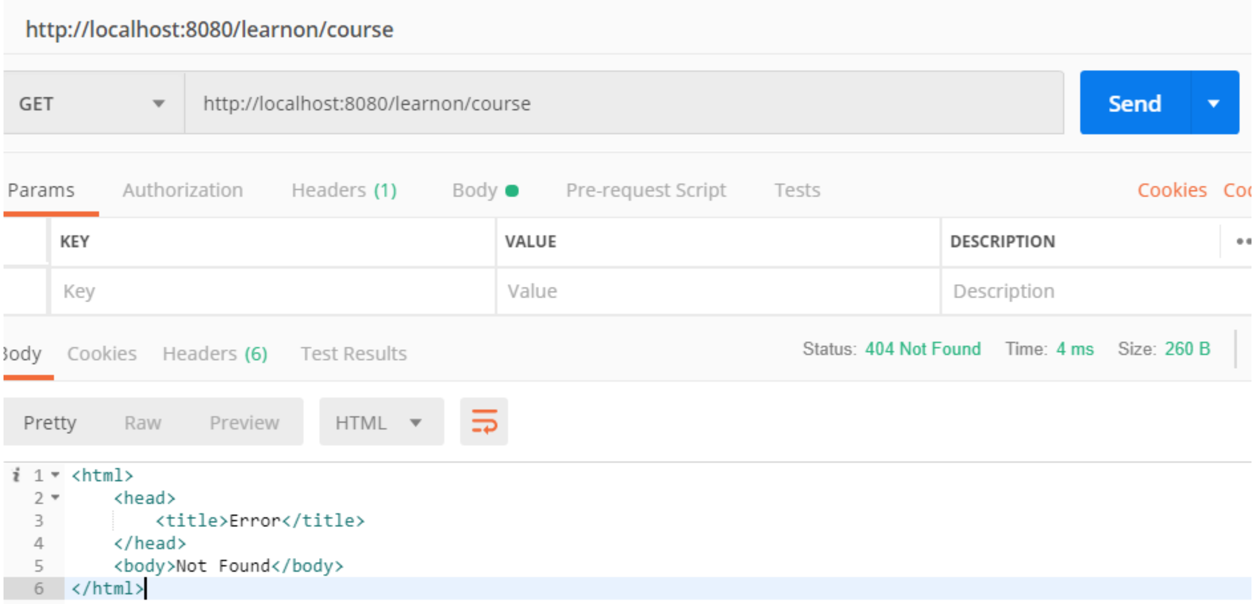
3. MANUAL TEST

MANUAL TEST RESULT							
TEST CASE ID	User Story ID	Scenario description	Scenario number	Test number	Date of Test	Status	Bug ID
1	All	Using an invalid URL	1	USallSC1	16 FEV	PASS	--
2	All	Database server is unavailable	2	USallSC2	16 FEV	PASS	--
3	All	Application server is unavailable	3	USallSC3	16 FEV	PASS	--
4	US1	Add a course successfully	1	US1SC1	24 FEV	PASS	--
5	US1	Method POST not exist for the path with id	2	US1SC2	24 FEV	PASS	--
6	US2	Retrieve data for all courses	1	US2SC1	24 FEV	PASS	--
7	US3	Retrieve data based on ID	1	US3SC1	24 FEV	PASS	--
8	US3	Inexistent ID is entered when retrieving a course	2	US3SC2	24 FEV	PASS	--
9	US4	Update data based on ID	1	US4SC1	24 FEV	PASS	--
10	US4	Update data based on inexistent ID	2	US4SC2	24 FEV	PASS	--
11	US4	Update data with path ID not matching entity ID	3	US4SC3	24 FEV	FAIL	US4 SC3 BI1
11	US4	Update data with path ID not matching entity ID	3	US4SC3	25 FEV	PASS	--
12	US4	Calling method PUT without passing ID	4	US4SC4	25 FEV		
13	US5	Delete data based on ID and confirm that is not possible to access to a deleted data	1	US5SC1	25 FEV	PASS	--
14	US5	Inexistent ID is entered when deleted a course	2	US5SC2	25 FEV	PASS	--

15	US5	Calling method DELETE without passing ID	3	US5SC3	25 FEV	PASS	--
16	US6	Add a user successfully	1	US6SC1	25 FEV	PASS	--
17	US6	Not possible to add a user with duplicate username	2	US6SC2	25 FEV	PASS	--
18	US6	Method POST not exist for the path with username	3	US6SC3	25 FEV	PASS	--
19	US7	Retrieve data for all users	1	US7SC1	25 FEV	PASS	--
20	US8	Retrieve data based on username	1	US8SC1	25 FEV	PASS	--
21	US8	Inexistent username is entered when retrieving a user	2	US8SC2	25 FEV	PASS	--
22	US9	Update data based on username	1	US9SC1	25 FEV	PASS	--
23	US9	Update data with path username not matching entity username	2	US9SC2	25 FEV	PASS	--
24	US9	Update data based on inexistent username	3	US9SC3	25 FEV	PASS	--
25	US9	Calling method PUT without passing username	4	US9SC4	25 FEV	PASS	--
26	US10	Delete data based on username and confirm that is not possible to access to a deleted data	1	US10SC1	25 FEV	PASS	--
27	US10	Inexistent username is entered when deleted a user	2	US10SC2	25 FEV	PASS	--
28	US10	Calling method DELETE without passing username	3	US10SC3	25 FEV	PASS	--

USER STORY ALL SCENARIO 1	General – Applies to all user stories
TEST CASE ID	1
DESCRIPTION	Using an invalid URL
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learnon/course Press send
THEN	HTTP Response 404 Not Found

TEST STATUS – PASSED



http://localhost:8080/learnon/course

GET http://localhost:8080/learnon/course

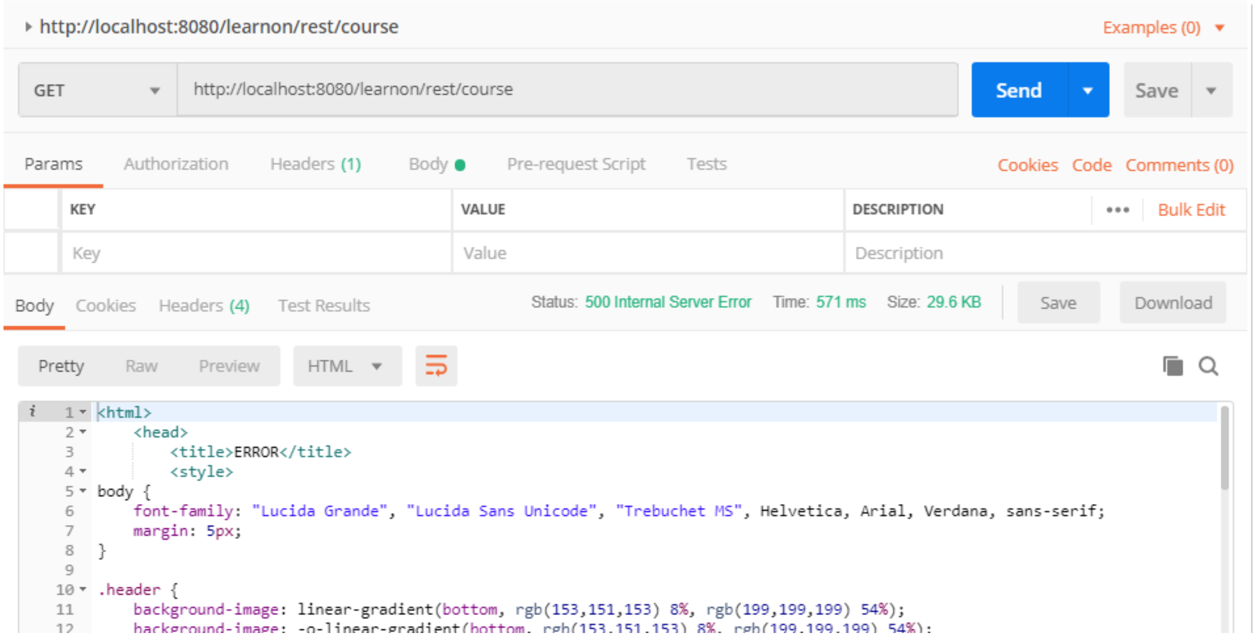
KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 404 Not Found Time: 4 ms Size: 260 B

```
<html>
    <head>
        <title>Error</title>
    </head>
    <body>Not Found</body>
</html>
```

USER STORY ALL SCENARIO 2	General – Applies to all user stories
TEST CASE ID	2
DESCRIPTION	Database server is unavailable
GIVEN	The Application Server is running The Database Server is stopped Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course Press send
THEN	HTTP Response 500 Internal Server Error

TEST STATUS – PASSED



▸ http://localhost:8080/learnon/rest/course

GET http://localhost:8080/learnon/rest/course

KEY	VALUE	DESCRIPTION
Key	Value	Description

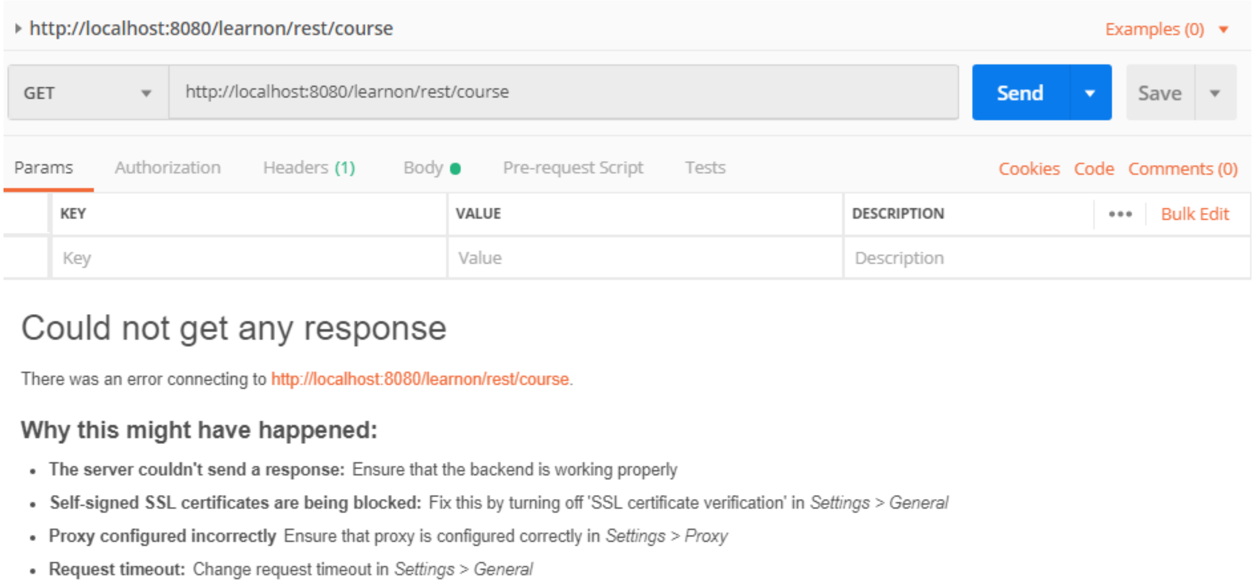
Status: 500 Internal Server Error Time: 571 ms Size: 29.6 KB

```
<html>
    <head>
        <title>ERROR</title>
        <style>
body {
    font-family: "Lucida Grande", "Lucida Sans Unicode", "Trebuchet MS", Helvetica, Arial, Verdana, sans-serif;
    margin: 5px;
}

.header {
    background-image: linear-gradient(bottom, rgb(153,151,153) 8%, rgb(199,199,199) 54%);
    background-image: -o-linear-gradient(bottom, rgb(153,151,153) 8%, rgb(199,199,199) 54%);
```

USER STORY ALL SCENARIO 3	General – Applies to all user stories
TEST CASE ID	3
DESCRIPTION	Application server is unavailable
GIVEN	The Application Server is stopped Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course Press send
THEN	HTTP Response Could not get any response

TEST STATUS – PASSED



http://localhost:8080/learnon/rest/course

KEY	VALUE	DESCRIPTION
Key	Value	Description

Could not get any response

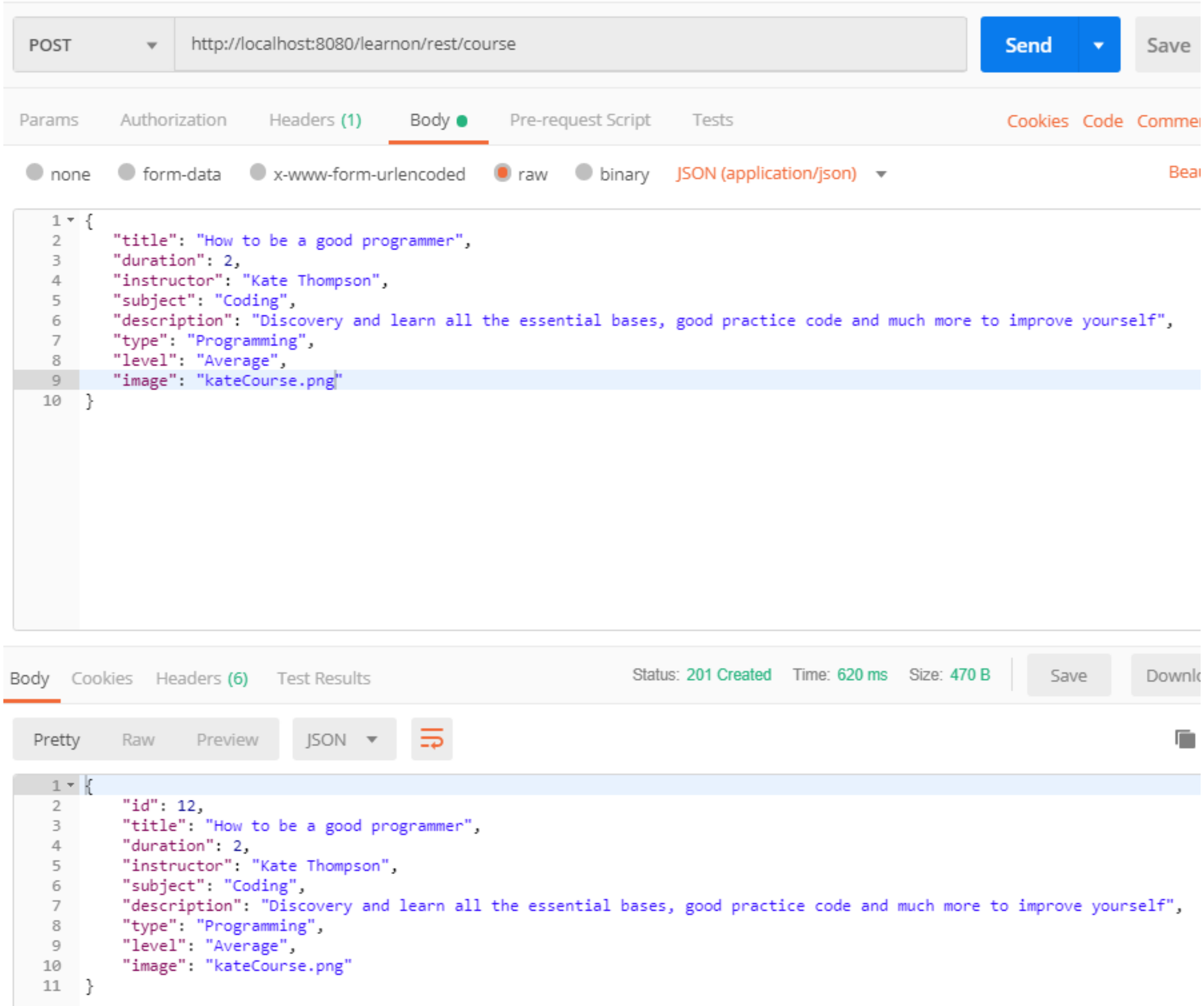
There was an error connecting to http://localhost:8080/learnon/rest/course.

Why this might have happened:

- **The server couldn't send a response:** Ensure that the backend is working properly
- **Self-signed SSL certificates are being blocked:** Fix this by turning off 'SSL certificate verification' in *Settings > General*
- **Proxy configured incorrectly** Ensure that proxy is configured correctly in *Settings > Proxy*
- **Request timeout:** Change request timeout in *Settings > General*

USER STORY 1 SCENARIO 1	As an API Consumer, I want to add a course, so that I can add a new course through my client application.
TEST CASE ID	4
DESCRIPTION	Add a course successfully
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set POST request Enter URL http://localhost:8080/learon/rest/course Press send
THEN	HTTP Response 201 CREATED Course is created and a JSON data format is generated.

TEST STATUS – PASSED



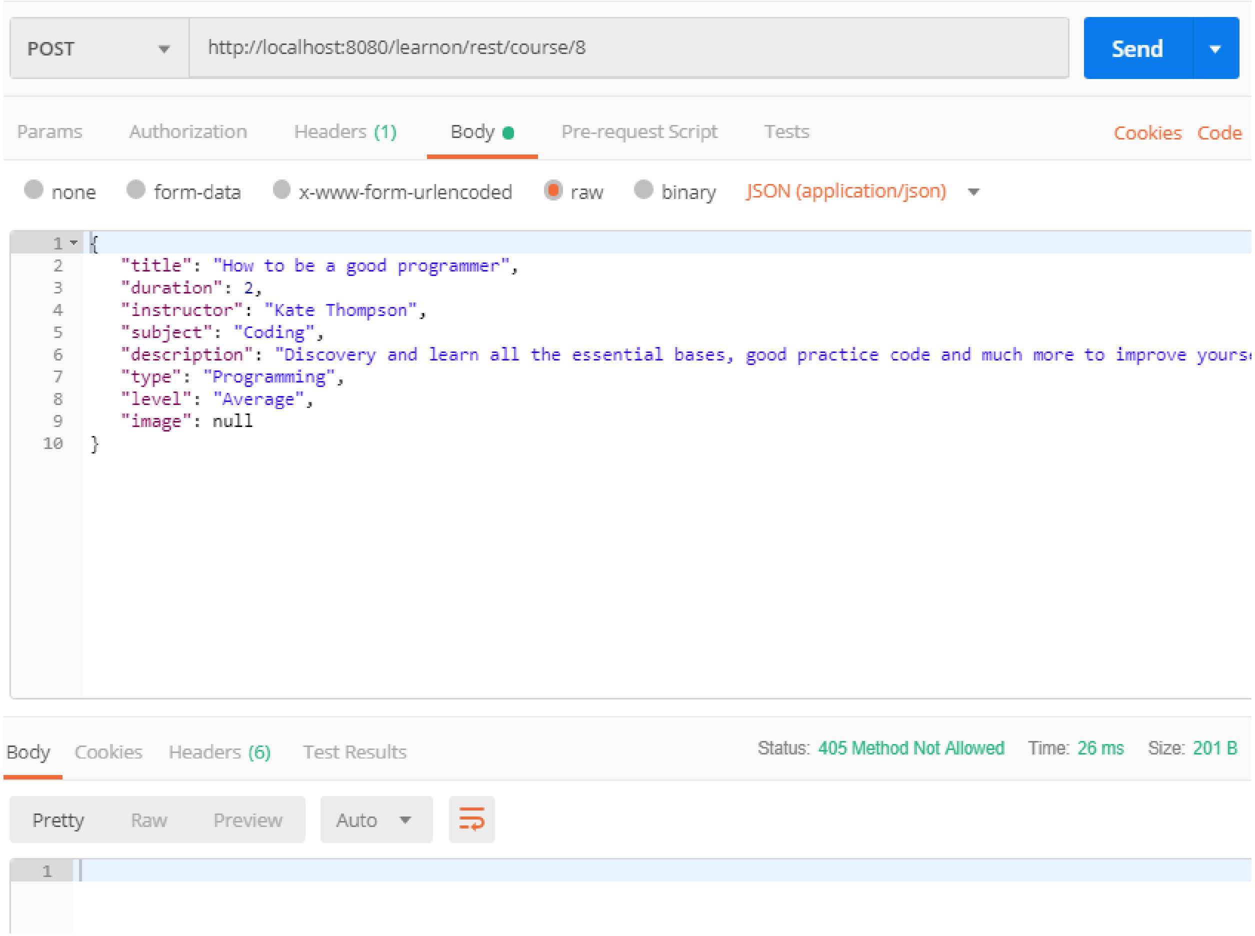
http://localhost:8080/learnon/rest/course

```
{
    "title": "How to be a good programmer",
    "duration": 2,
    "instructor": "Kate Thompson",
    "subject": "Coding",
    "description": "Discovery and learn all the essential bases, good practice code and much more to improve yourself",
    "type": "Programming",
    "level": "Average",
    "image": "kateCourse.png"
}
```

Status: 201 Created Time: 620 ms Size: 470 B

```
{
    "id": 12,
    "title": "How to be a good programmer",
    "duration": 2,
    "instructor": "Kate Thompson",
    "subject": "Coding",
    "description": "Discovery and learn all the essential bases, good practice code and much more to improve yourself",
    "type": "Programming",
    "level": "Average",
    "image": "kateCourse.png"
}
```

USER STORY 1 SCENARIO 2	As an API Consumer, I want to add a course, so that I can add a new course through my client application.
TEST CASE ID	5
DESCRIPTION	Method POST not exist for the path with id
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set POST request Enter URL http://localhost:8080/learon/rest/course/{id} Press send
THEN	HTTP Response 405 METHOD NOT ALLOWED Course is not created.
TEST STATUS – PASSED	



POST http://localhost:8080/learnon/rest/course/8

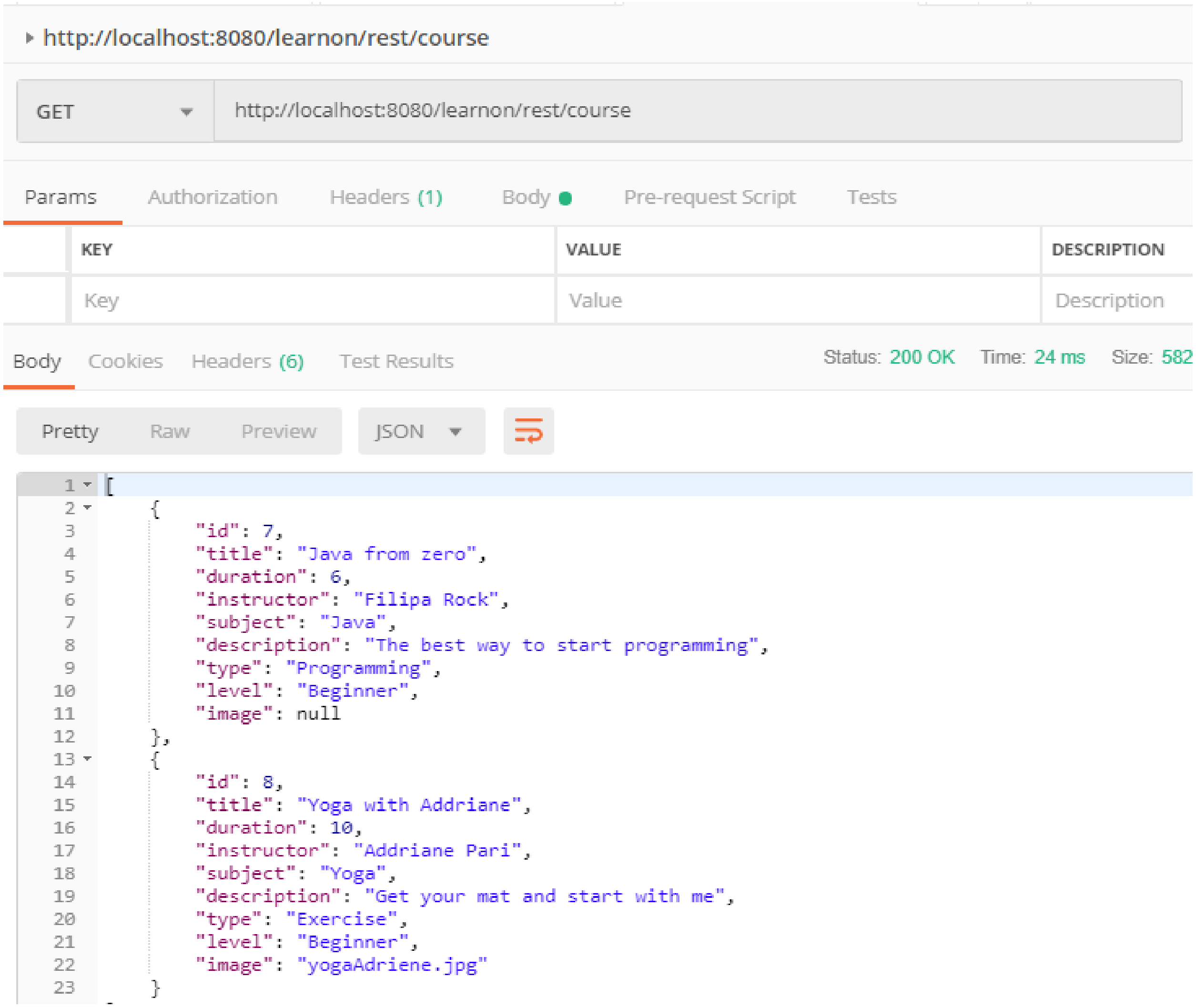
Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary JSON (application/json)

```
{
    "title": "How to be a good programmer",
    "duration": 2,
    "instructor": "Kate Thompson",
    "subject": "Coding",
    "description": "Discovery and learn all the essential bases, good practice code and much more to improve yours
    "type": "Programming",
    "level": "Average",
    "image": null
}
```

Status: 405 Method Not Allowed Time: 26 ms Size: 201 B

USER STORY 2 SCENARIO 1	As an API Consumer, I want to retrieve data in JSON format for all courses, so that I can use the data in my client application.
TEST CASE ID	6
DESCRIPTION	Retrieve data for all courses
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course Press send
THEN	HTTP Response 200 OK All database entries are displayed in JSON format

TEST STATUS – PASSED



http://localhost:8080/learnon/rest/course

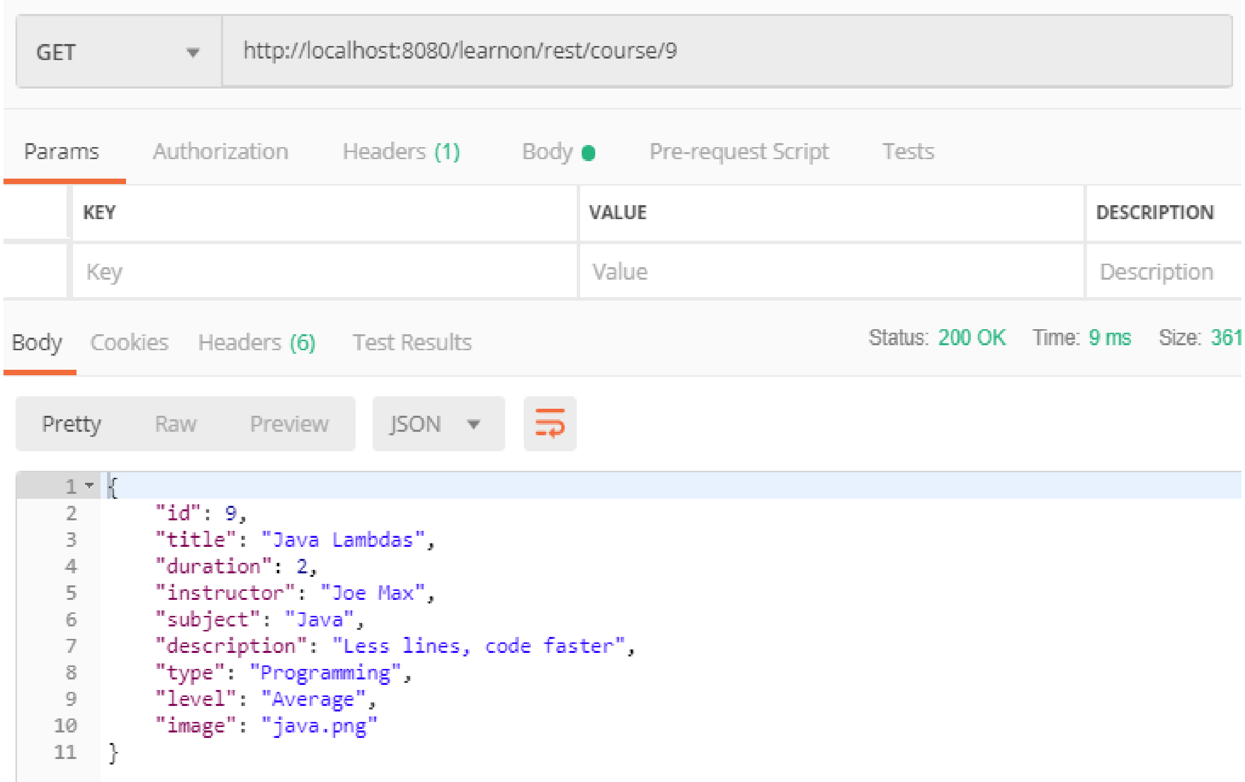
GET http://localhost:8080/learnon/rest/course

Status: 200 OK Time: 24 ms Size: 582

```
[
    {
        "id": 7,
        "title": "Java from zero",
        "duration": 6,
        "instructor": "Filipa Rock",
        "subject": "Java",
        "description": "The best way to start programming",
        "type": "Programming",
        "level": "Beginner",
        "image": null
    },
    {
        "id": 8,
        "title": "Yoga with Addriane",
        "duration": 10,
        "instructor": "Addriane Pari",
        "subject": "Yoga",
        "description": "Get your mat and start with me",
        "type": "Exercise",
        "level": "Beginner",
        "image": "yogaAdriene.jpg"
    }
```

USER STORY 3 SCENARIO 1	As an API Consumer, I want to retrieve data in JSON format for one course based on its ID, so that I can use the data in my client application.
TEST CASE ID	7
DESCRIPTION	Retrieve data based on ID
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course/{valid id} Press send
THEN	HTTP Response 200 OK Details for course displayed in JSON format corresponding to entered ID

TEST STATUS – PASSED



GET http://localhost:8080/learnon/rest/course/9

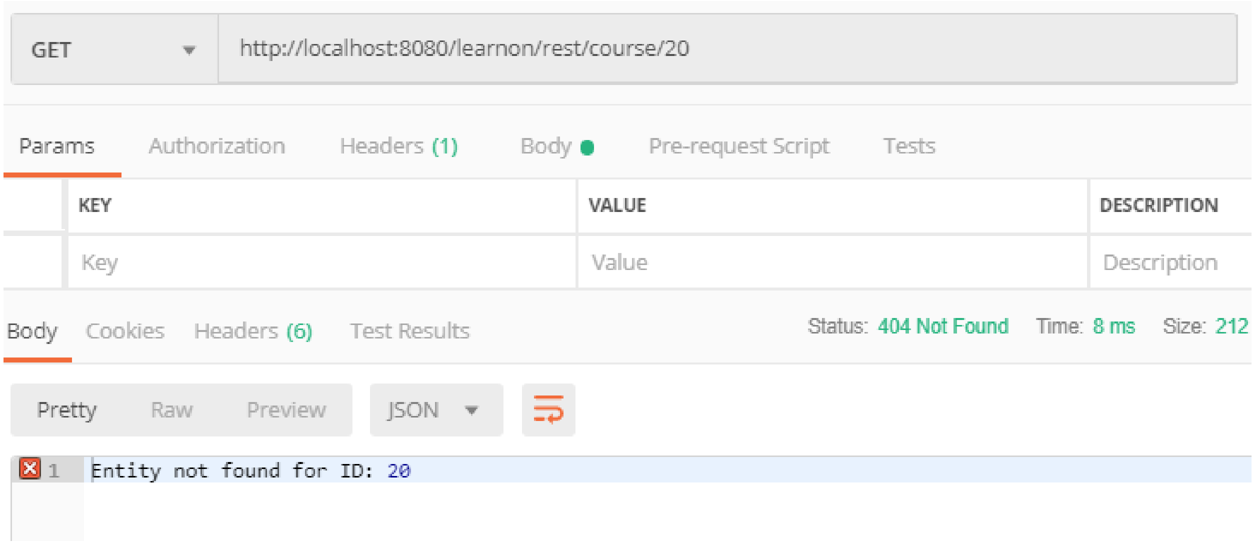
Status: 200 OK Time: 9 ms Size: 361

JSON

```
{
    "id": 9,
    "title": "Java Lambdas",
    "duration": 2,
    "instructor": "Joe Max",
    "subject": "Java",
    "description": "Less lines, code faster",
    "type": "Programming",
    "level": "Average",
    "image": "java.png"
}
```

USER STORY 3 SCENARIO 2	As an API Consumer, I want to retrieve data in JSON format for one course based on its ID, so that I can use the data in my client application.
TEST CASE ID	8
DESCRIPTION	Inexistent ID is entered when retrieving a course - success
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course/{inexistent id} Press send
THEN	HTTP Response 404 NOT FOUND Entity not found for ID: {inexistent id}

TEST STATUS – PASSED



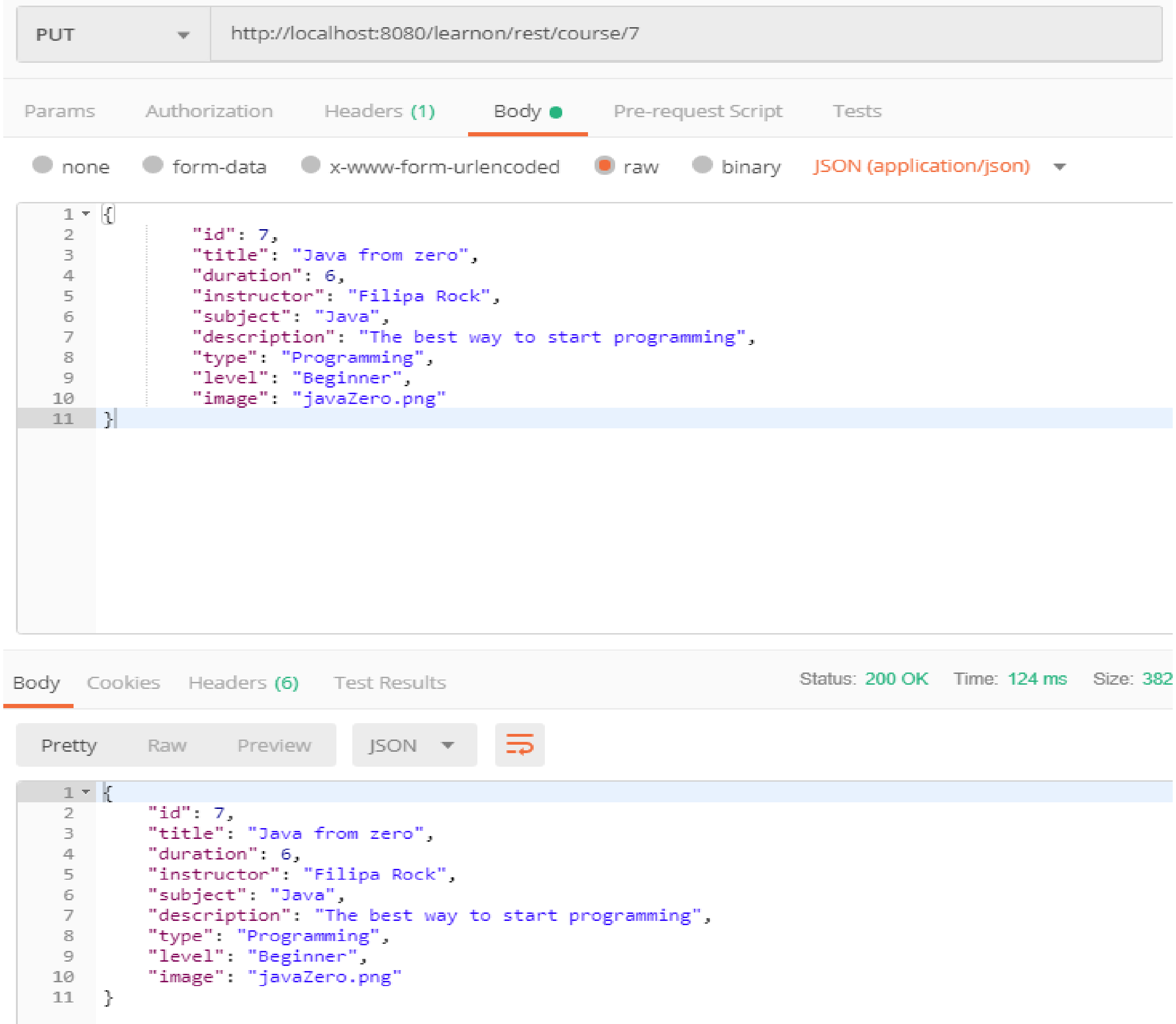
GET http://localhost:8080/learnon/rest/course/20

Status: 404 Not Found Time: 8 ms Size: 212

```
Entity not found for ID: 20
```

USER STORY 4 SCENARIO 1	As an API Consumer, I want to update a course, so that I can update an entry through my client application.
TEST CASE ID	9
DESCRIPTION	Update data based on ID
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/course/{valid id} Press send
THEN	HTTP Response 200 OK Course to the corresponding entered ID is updated, so results a JSON data to the updated data.

TEST STATUS – PASSED



PUT http://localhost:8080/learnon/rest/course/7

● raw JSON (application/json)

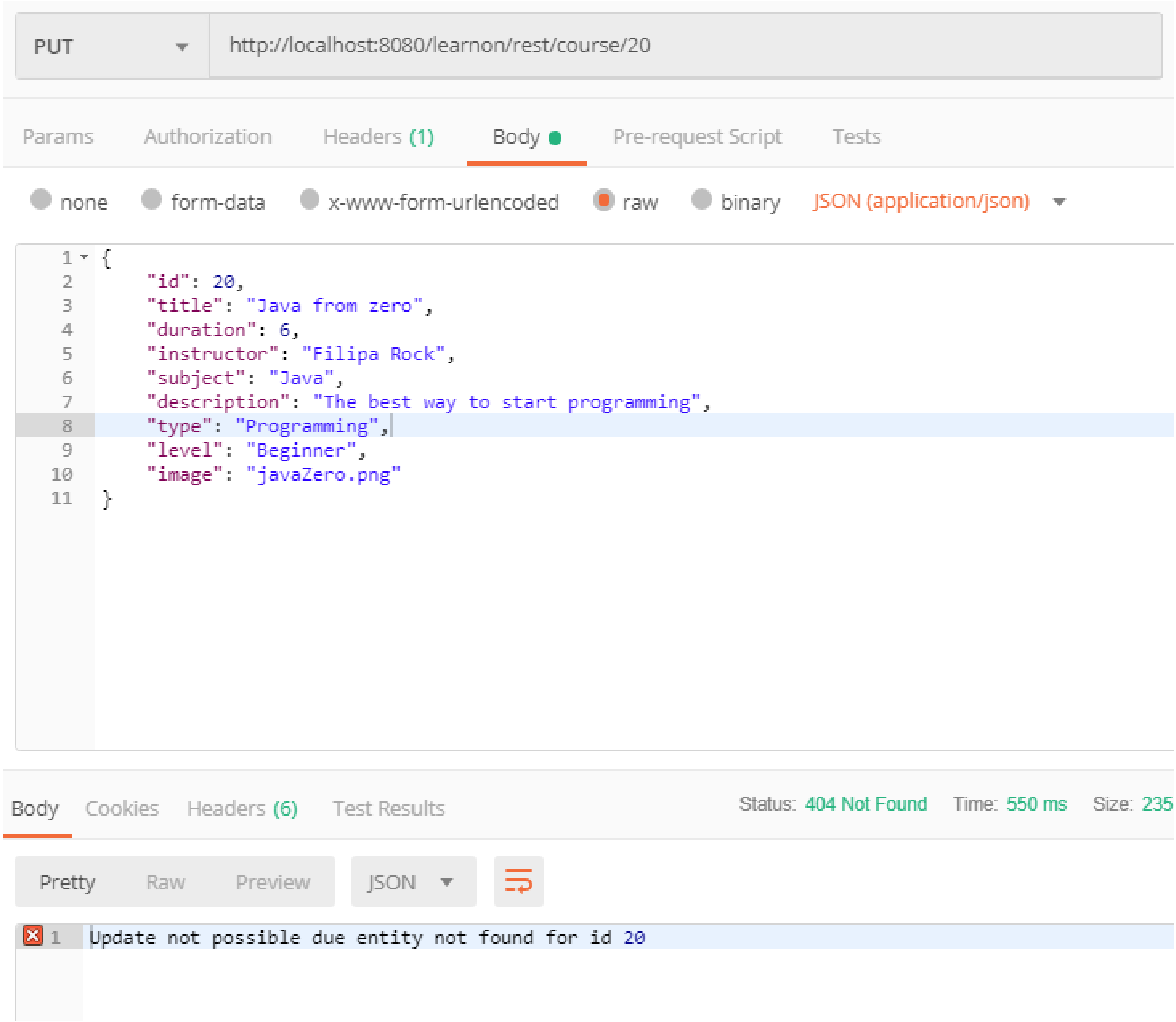
```
{
    "id": 7,
    "title": "Java from zero",
    "duration": 6,
    "instructor": "Filipa Rock",
    "subject": "Java",
    "description": "The best way to start programming",
    "type": "Programming",
    "level": "Beginner",
    "image": "javaZero.png"
}
```

Status: 200 OK Time: 124 ms Size: 382

```
{
  "id": 7,
  "title": "Java from zero",
  "duration": 6,
  "instructor": "Filipa Rock",
  "subject": "Java",
  "description": "The best way to start programming",
  "type": "Programming",
  "level": "Beginner",
  "image": "javaZero.png"
}
```

USER STORY 4 SCENARIO 2	As an API Consumer, I want to update a course, so that I can update an entry through my client application.
TEST CASE ID	10
DESCRIPTION	Update data based on inexistent ID
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/course/{inexistent id} Press send
THEN	HTTP Response 404 NOT FOUND Update not possible due entity not found for id: {inexistent id}

TEST STATUS – PASSED



http://localhost:8080/learnon/rest/course/20

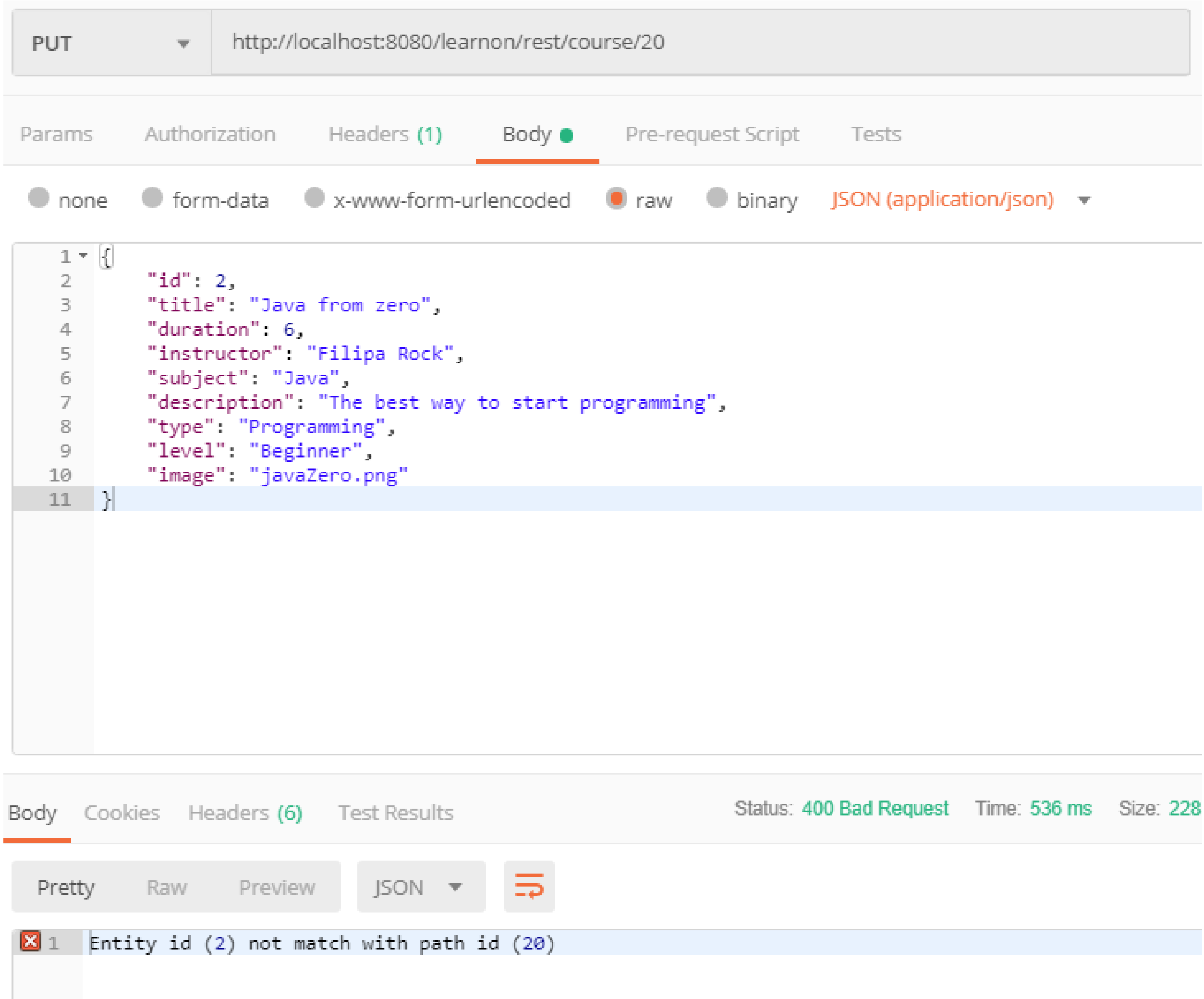
```
{
    "id": 20,
    "title": "Java from zero",
    "duration": 6,
    "instructor": "Filipa Rock",
    "subject": "Java",
    "description": "The best way to start programming",
    "type": "Programming",
    "level": "Beginner",
    "image": "javaZero.png"
}
```

Status: 404 Not Found Time: 550 ms Size: 235

```
Update not possible due entity not found for id 20
```

USER STORY 4 SCENARIO 3	As an API Consumer, I want to update a course, so that I can update an entry through my client application.
TEST CASE ID	11
DESCRIPTION	Update data with path ID not matching entity ID
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/course/{inexistent id} Press send
THEN	HTTP Response 400 BAD REQUEST Entity id (id number) not match with path id (id number).

TEST STATUS – PASSED



PUT http://localhost:8080/learnon/rest/course/20

● raw JSON (application/json)

```
{
    "id": 2,
    "title": "Java from zero",
    "duration": 6,
    "instructor": "Filipa Rock",
    "subject": "Java",
    "description": "The best way to start programming",
    "type": "Programming",
    "level": "Beginner",
    "image": "javaZero.png"
}
```

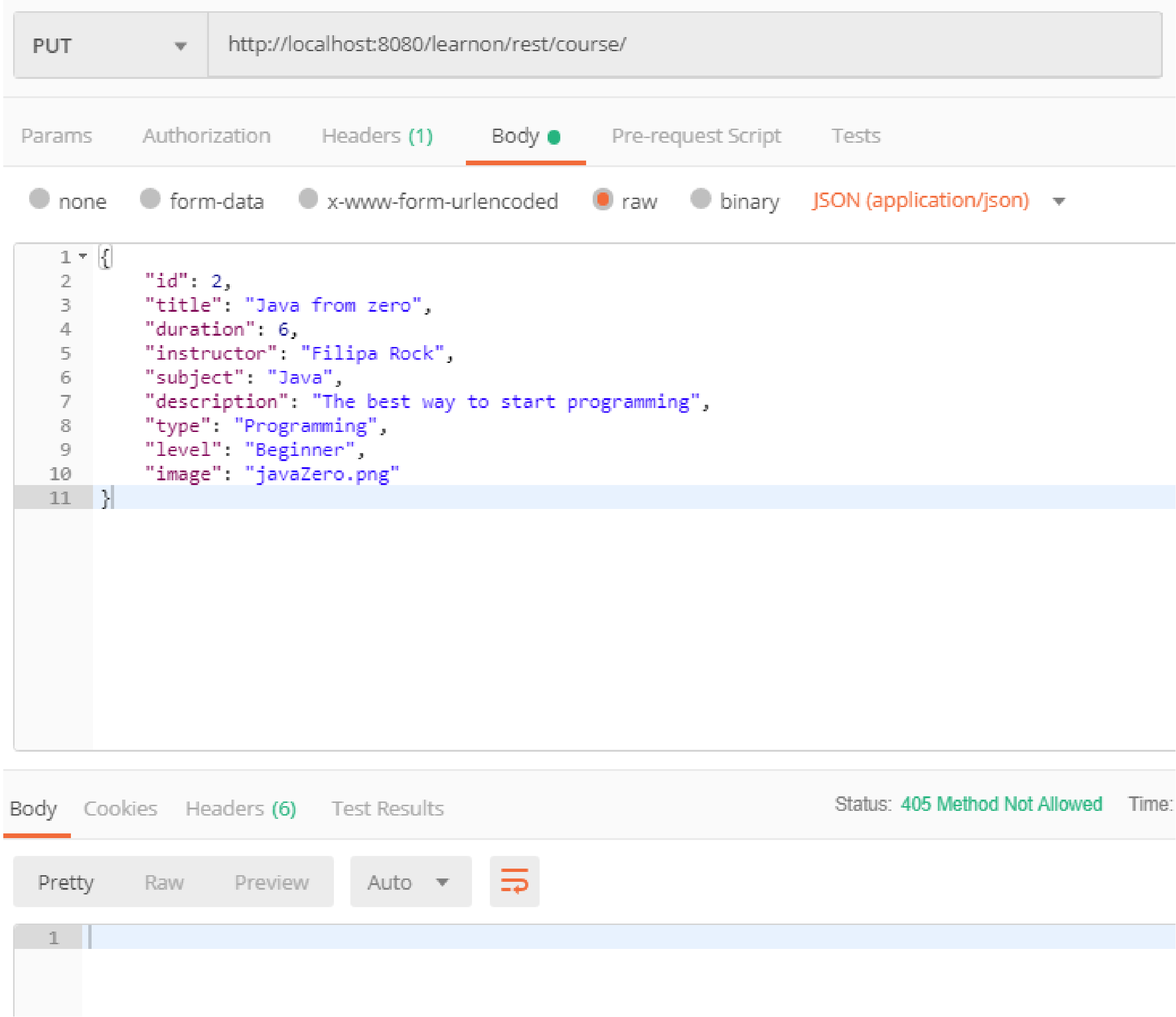
Status: 400 Bad Request Time: 536 ms Size: 228

JSON

```
Entity id (2) not match with path id (20)
```

USER STORY 4 SCENARIO 4	As an API Consumer, I want to update a course, so that I can update an entry through my client application.
TEST CASE ID	12
DESCRIPTION	Calling method PUT without passing ID
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/course/ Press send
THEN	HTTP Response 405 METHOD NOT ALLOWED No update performed

TEST STATUS – PASSED



PUT http://localhost:8080/learnon/rest/course/

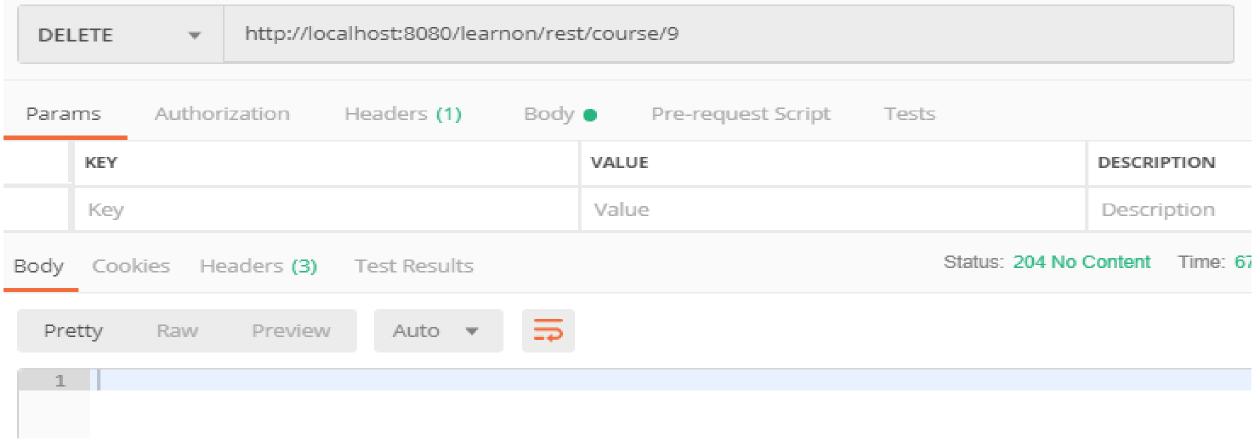
Body: raw, JSON (application/json)

```
{
    "id": 2,
    "title": "Java from zero",
    "duration": 6,
    "instructor": "Filipa Rock",
    "subject": "Java",
    "description": "The best way to start programming",
    "type": "Programming",
    "level": "Beginner",
    "image": "javaZero.png"
}
```

Status: 405 Method Not Allowed Time:

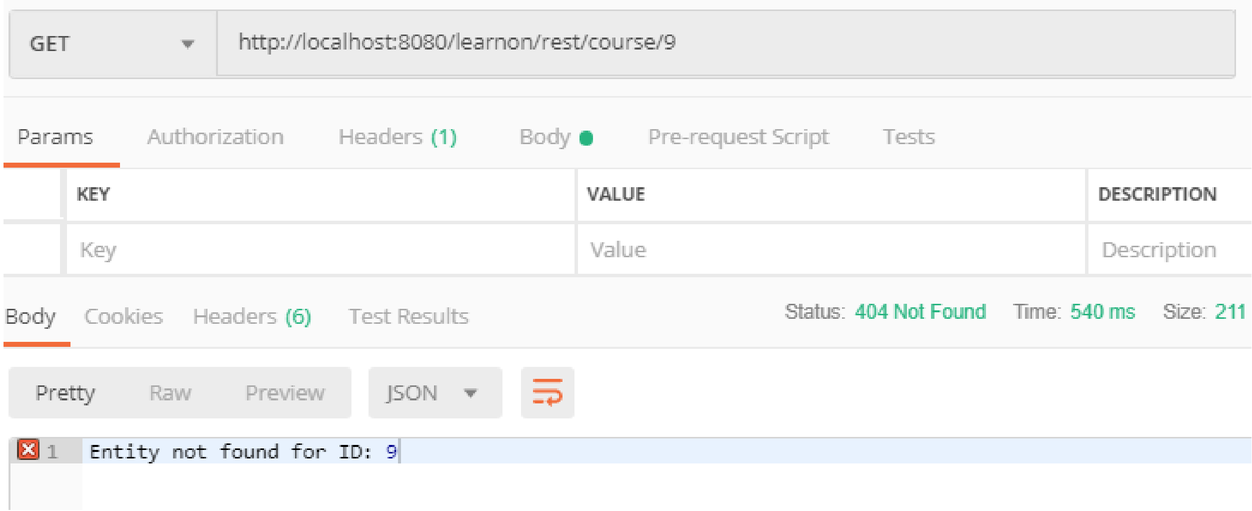
USER STORY 5 SCENARIO 1	As an API Consumer, I want to delete a course, so that I can remove unwanted entries through my client application.
TEST CASE ID	13
DESCRIPTION	Delete data based on ID and confirm that is not possible to access to a deleted data
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set DELETE request Enter URL http://localhost:8080/learon/rest/course/{valid id} Press send
THEN	HTTP Response 204 NO CONTENT The data is deleted successful so no content is showed.
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course/{deleted id} Press send
THEN	HTTP Response 404 NOT FOUND Entity not found for ID: {deleted id}

TEST STATUS – PASSED


DELETE http://localhost:8080/learnon/rest/course/9

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 204 No Content Time: 67


GET http://localhost:8080/learnon/rest/course/9

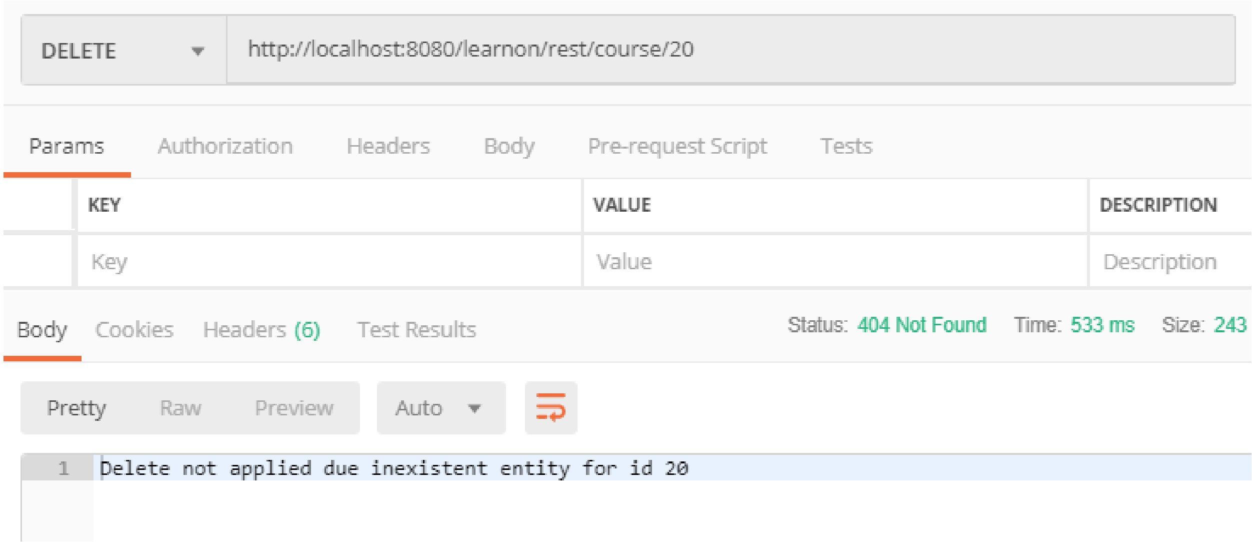
KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 404 Not Found Time: 540 ms Size: 211

```
Entity not found for ID: 9
```

USER STORY 5 SCENARIO 2	As an API Consumer, I want to delete a course, so that I can remove unwanted entries through my client application.
TEST CASE ID	14
DESCRIPTION	Inexistent ID is entered when deleted a course
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course/{inexistend id} Press send
THEN	HTTP Response 404 NOT FOUND Delete not applied due inexistent entity for id {inexistent id}

TEST STATUS – PASSED



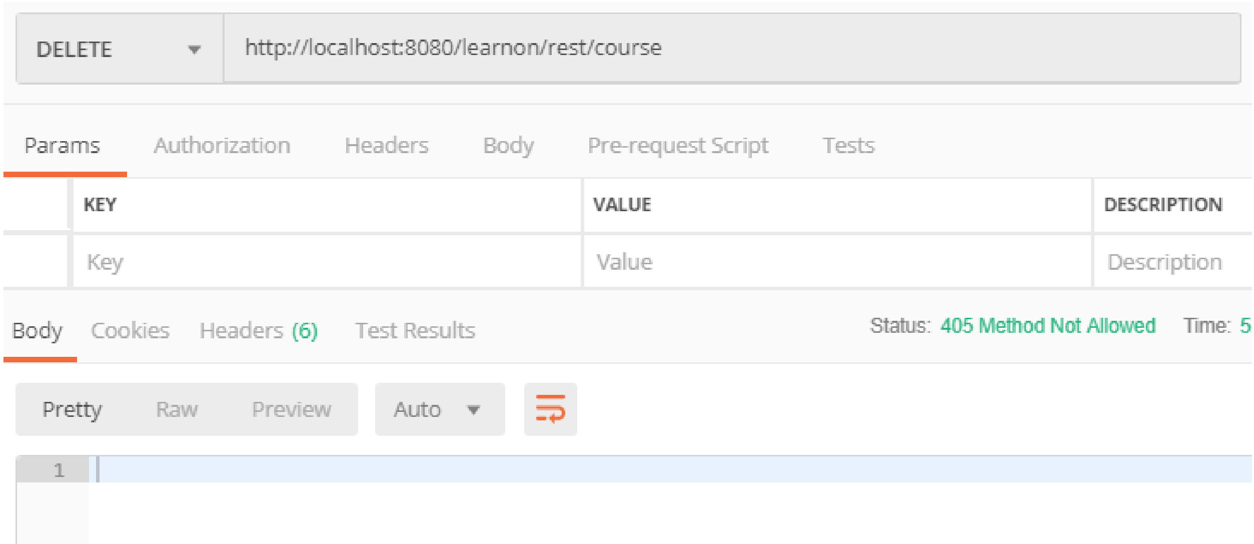
DELETE http://localhost:8080/learnon/rest/course/20

Status: 404 Not Found Time: 533 ms Size: 243

```
Delete not applied due inexistent entity for id 20
```

USER STORY 5 SCENARIO 3	As an API Consumer, I want to delete a course, so that I can remove unwanted entries through my client application.
TEST CASE ID	15
DESCRIPTION	Calling method DELETE without passing ID
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/course Press send
THEN	HTTP Response 405 METHOD NOT ALLOWED No delete is perfomed

TEST STATUS – PASSED



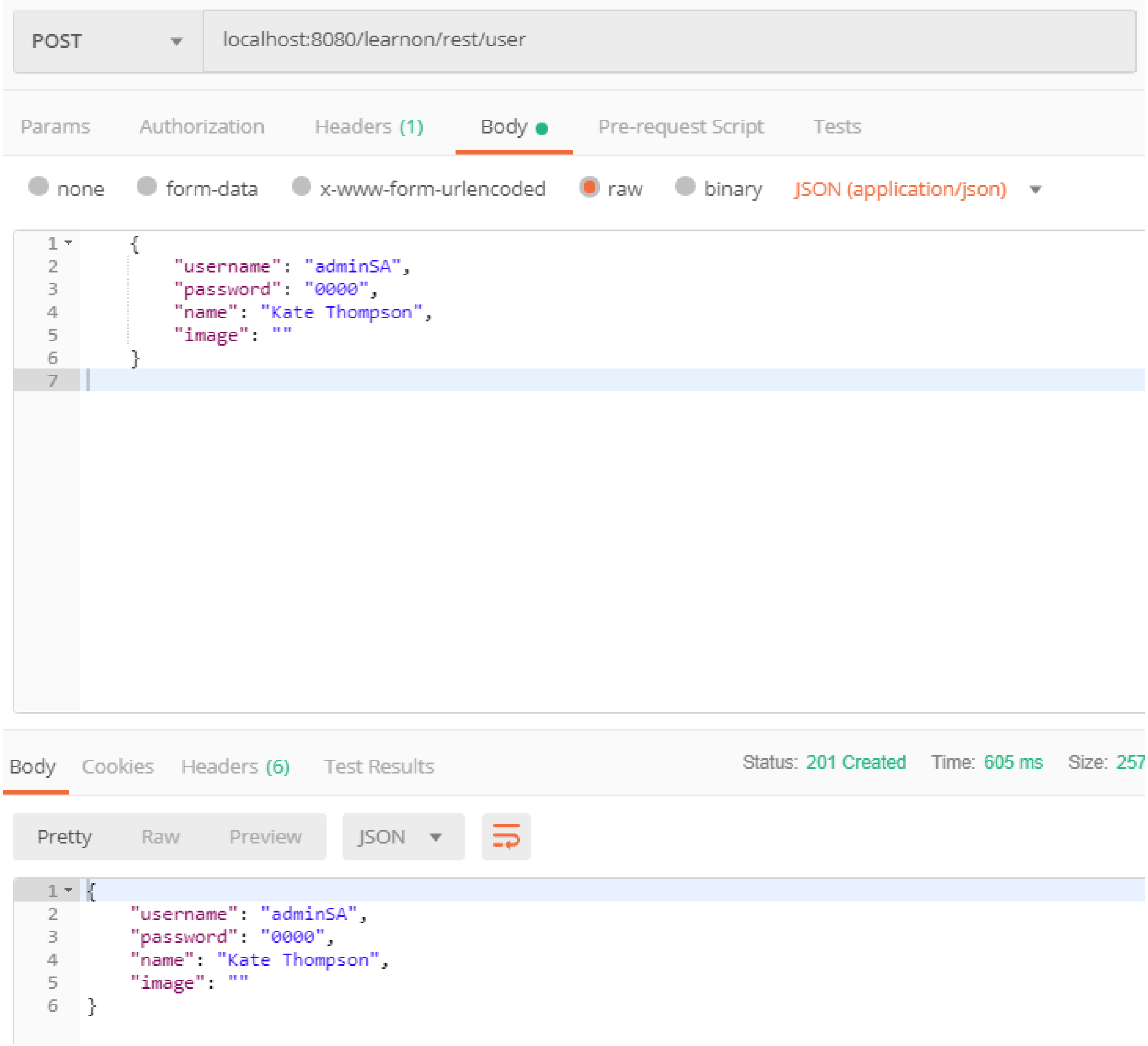
DELETE http://localhost:8080/learnon/rest/course

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 405 Method Not Allowed Time: 5

USER STORY 6 SCENARIO 1	As an API Consumer, I want to add a user, so that I can add a new user through my client application.
TEST CASE ID	16
DESCRIPTION	Add a user successfully
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set POST request Enter URL http://localhost:8080/learon/rest/user Press send
THEN	HTTP Response 201 CREATED User is created and a JSON data format is generated.

TEST STATUS – PASSED



POST ▾ localhost:8080/learnon/rest/user

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary JSON (application/json) ▾

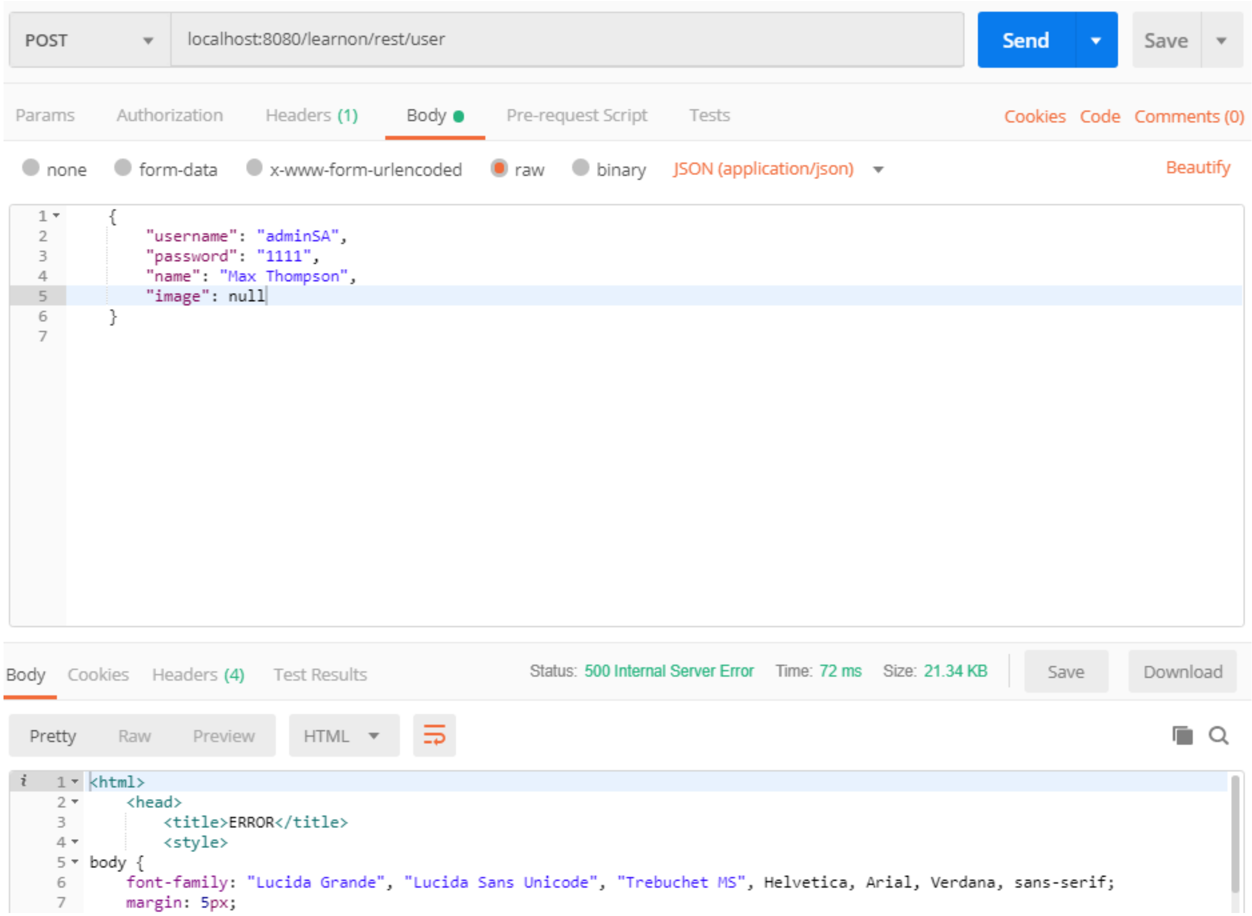
```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate Thompson",
    "image": ""
}
```

Status: 201 Created Time: 605 ms Size: 257

```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate Thompson",
    "image": ""
}
```

USER STORY 6 SCENARIO 2	As an API Consumer, I want to add a user, so that I can add a new user through my client application.
TEST CASE ID	17
DESCRIPTION	Not possible to add a user with duplicate username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set POST request Enter URL http://localhost:8080/learon/rest/user Press send
THEN	HTTP Response 500 INTERNAL SERVER ERROR No user is created.

TEST STATUS – PASSED



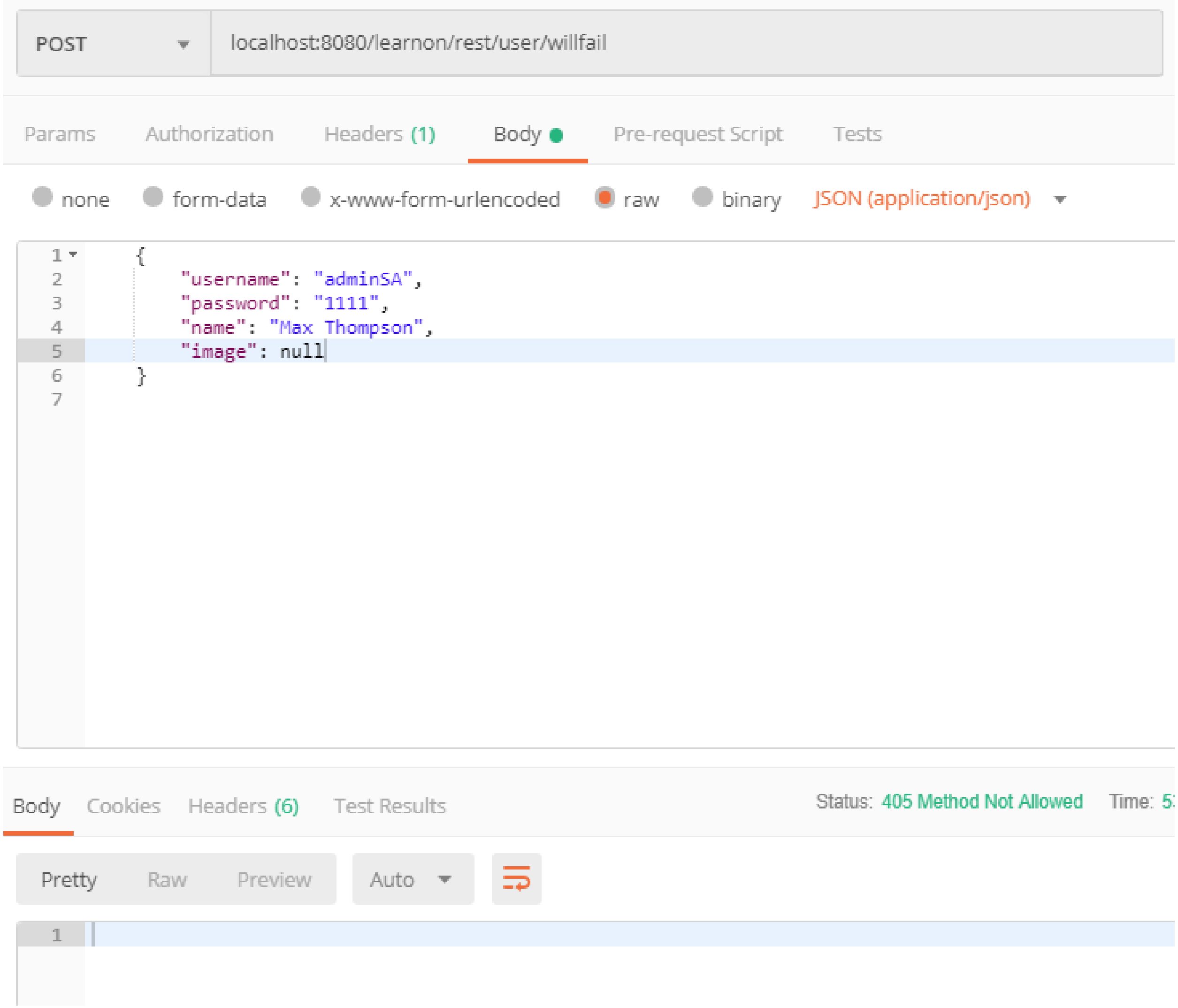
```
{
    "username": "adminSA",
    "password": "1111",
    "name": "Max Thompson",
    "image": null
}
```

Status: 500 Internal Server Error Time: 72 ms Size: 21.34 KB

```
<html>
    <head>
        <title>ERROR</title>
        <style>
body {
    font-family: "Lucida Grande", "Lucida Sans Unicode", "Trebuchet MS", Helvetica, Arial, Verdana, sans-serif;
    margin: 5px;
```

USER STORY 6 SCENARIO 3	As an API Consumer, I want to add a user, so that I can add a new user through my client application.
TEST CASE ID	18
DESCRIPTION	Method POST not exist for the path with username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set POST request Enter URL http://localhost:8080/learon/rest/user/{username} Press send
THEN	HTTP Response 405 METHOD NOT ALLOWED No user is created.

TEST STATUS – PASSED



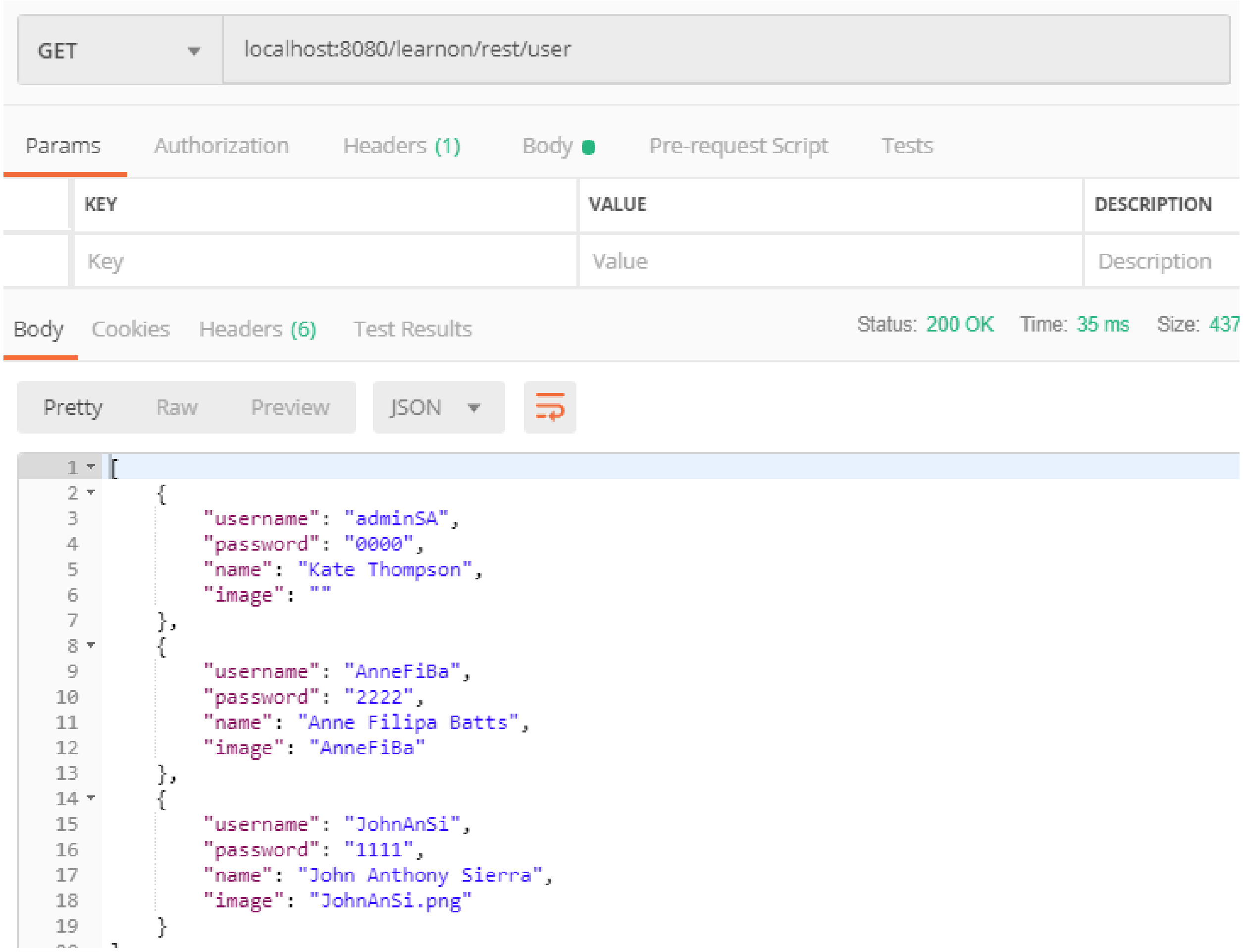
POST localhost:8080/learnon/rest/user/willfail

Body: raw, JSON (application/json)

```
{
    "username": "adminSA",
    "password": "1111",
    "name": "Max Thompson",
    "image": null
}
```

Status: 405 Method Not Allowed Time: 5

USER STORY 7 SCENARIO 1	As an API Consumer, I want to retrieve data in JSON format for all users, so that I can use the data in my client application.
TEST CASE ID	19
DESCRIPTION	Retrieve data for all users
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/user Press send
THEN	HTTP Response 200 OK All database entries are displayed in JSON format
TEST STATUS – PASSED	

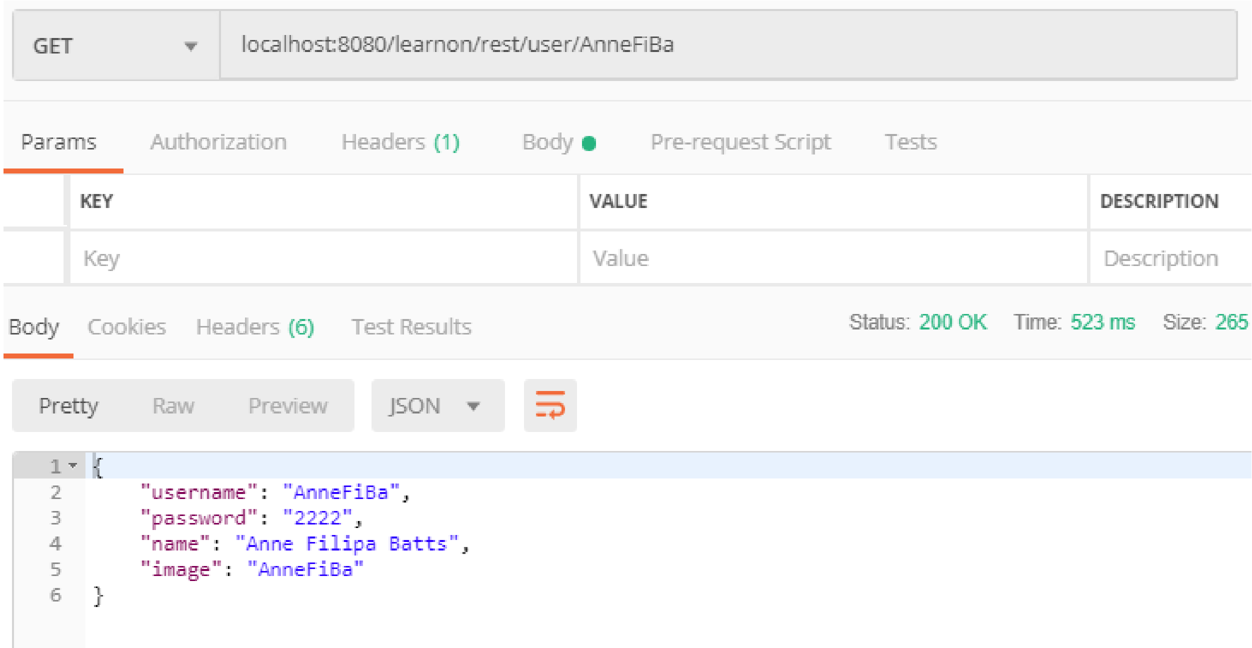


localhost:8080/learnon/rest/user

Status: 200 OK Time: 35 ms Size: 437

```
[
    {
        "username": "adminSA",
        "password": "0000",
        "name": "Kate Thompson",
        "image": ""
    },
    {
        "username": "AnneFiBa",
        "password": "2222",
        "name": "Anne Filipa Batts",
        "image": "AnneFiBa"
    },
    {
        "username": "JohnAnSi",
        "password": "1111",
        "name": "John Anthony Sierra",
        "image": "JohnAnSi.png"
    }
```

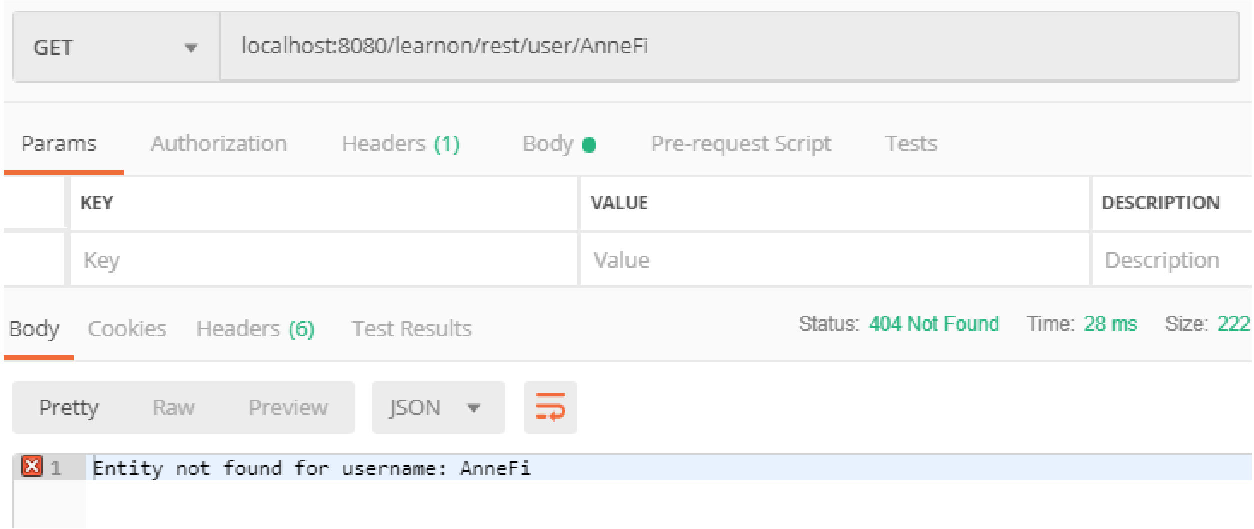
USER STORY 8 SCENARIO 1	As an API Consumer, I want to retrieve data in JSON format for one user based on its username, so that I can use the data in my client application.
TEST CASE ID	20
DESCRIPTION	Retrieve data based on username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/user/{valid username} Press send
THEN	HTTP Response 200 OK Details for user displayed in JSON format corresponding to entered username
TEST STATUS – PASSED	



```
{
    "username": "AnneFiBa",
    "password": "2222",
    "name": "Anne Filipa Batts",
    "image": "AnneFiBa"
}
```

USER STORY 8 SCENARIO 2	As an API Consumer, I want to retrieve data in JSON format for one user based on its username, so that I can use the data in my client application.
TEST CASE ID	21
DESCRIPTION	Inexistent username is entered when retrieving a user
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/user/{inexistent username} Press send
THEN	HTTP Response 404 NOT FOUND Entity not found for username: {inexistent username}

TEST STATUS – PASSED



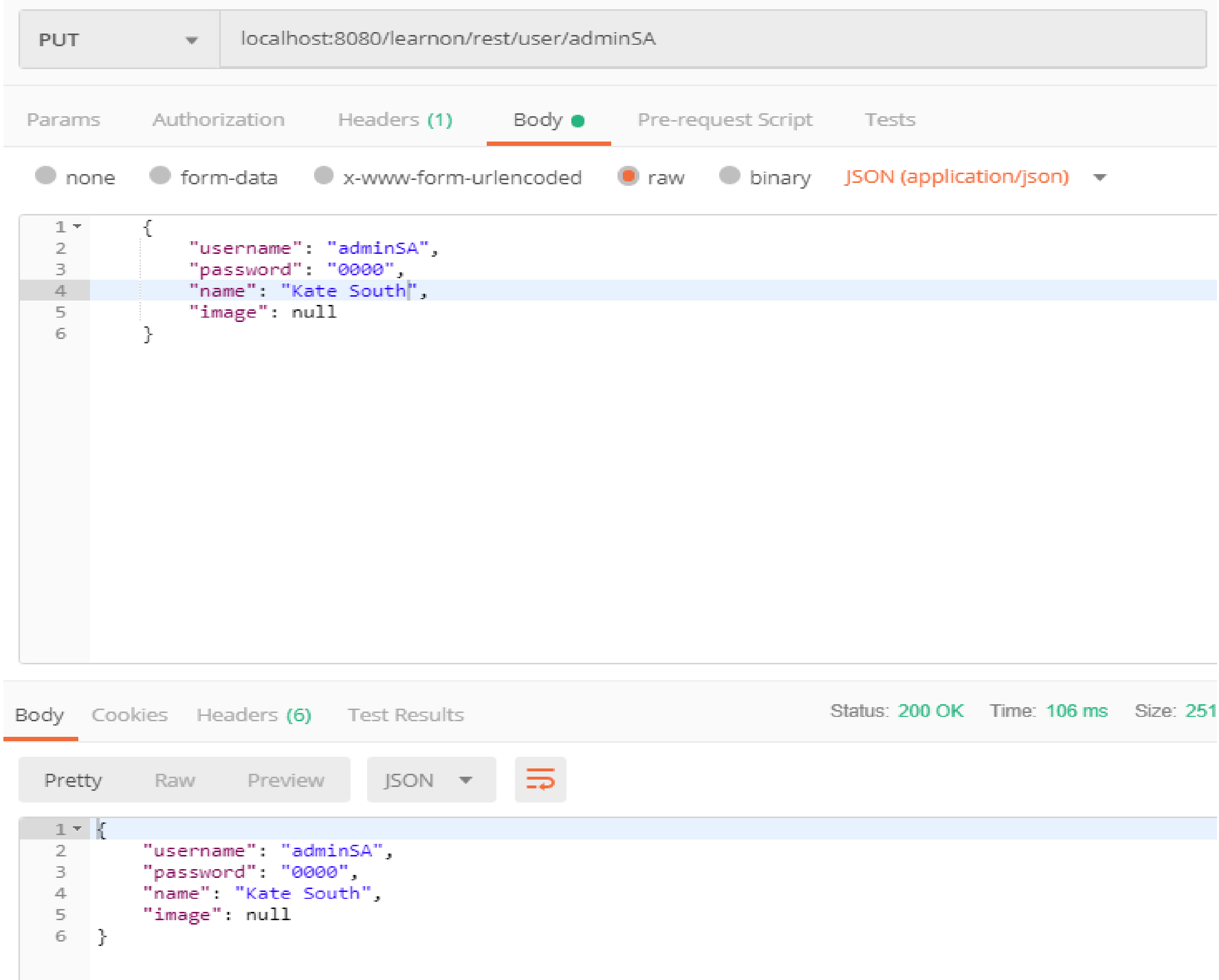
GET localhost:8080/learnon/rest/user/AnneFi

Status: 404 Not Found Time: 28 ms Size: 222

```
Entity not found for username: AnneFi
```

USER STORY 9 SCENARIO 1	As an API Consumer, I want to update a user, so that I can update an entry through my client application.
TEST CASE ID	22
DESCRIPTION	Update data based on username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/user/{valid username} Press send
THEN	HTTP Response 200 OK User to the corresponding entered username is updated, so results a JSON data to the updated data.

TEST STATUS – PASSED



PUT localhost:8080/learnon/rest/user/adminSA

Body: raw JSON (application/json)

```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate South",
    "image": null
}
```

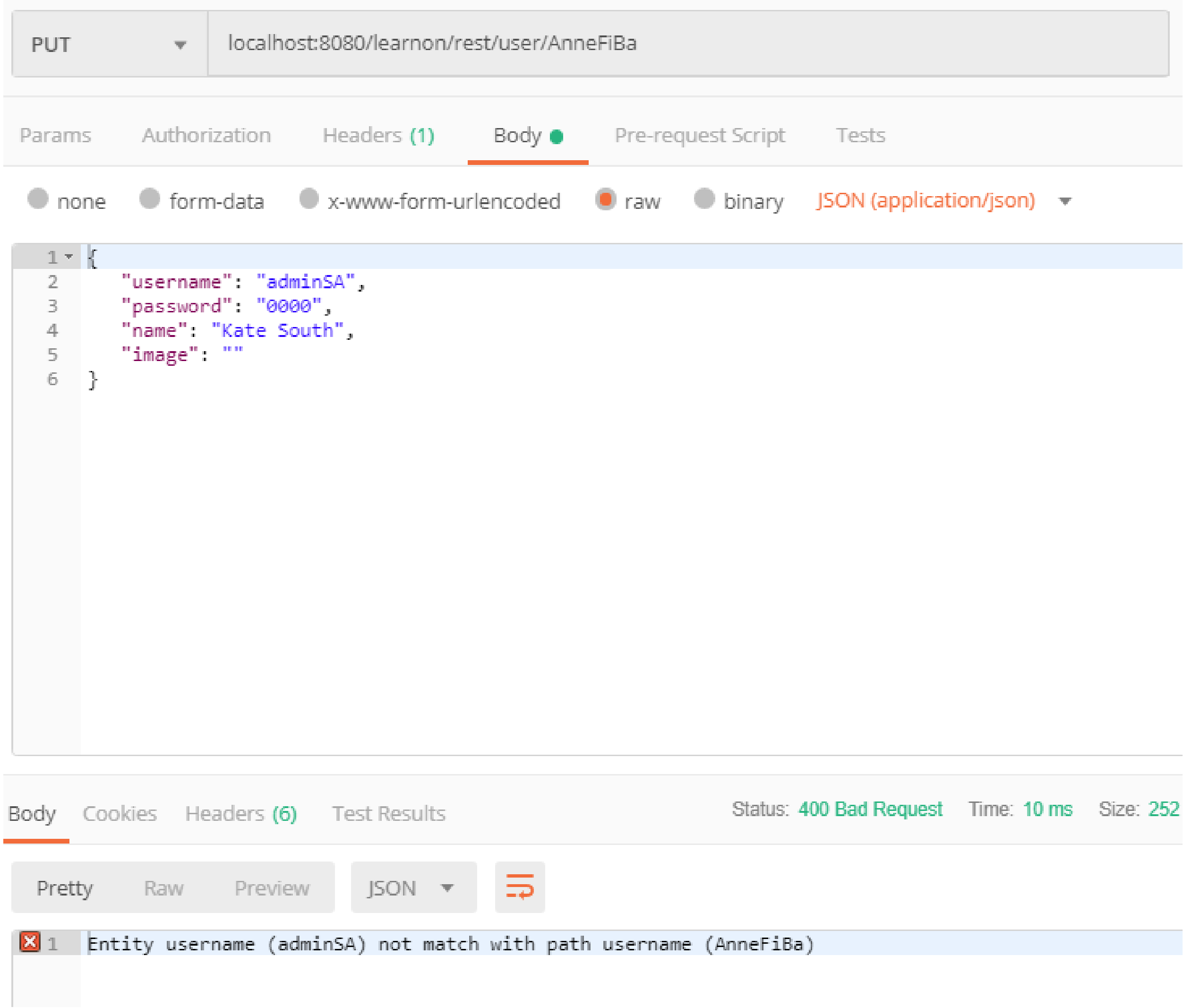
Status: 200 OK Time: 106 ms Size: 251

Body JSON

```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate South",
    "image": null
}
```

USER STORY 9 SCENARIO 2	As an API Consumer, I want to update a user, so that I can update an entry through my client application.
TEST CASE ID	23
DESCRIPTION	Update data with path username not matching entity username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/user/{username} Press send
THEN	HTTP Response 400 BAD REQUEST Entity username (username value) not match with path username (username value)

TEST STATUS – PASSED



PUT localhost:8080/learnon/rest/user/AnneFiBa

● raw JSON (application/json)

```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate South",
    "image": ""
}
```

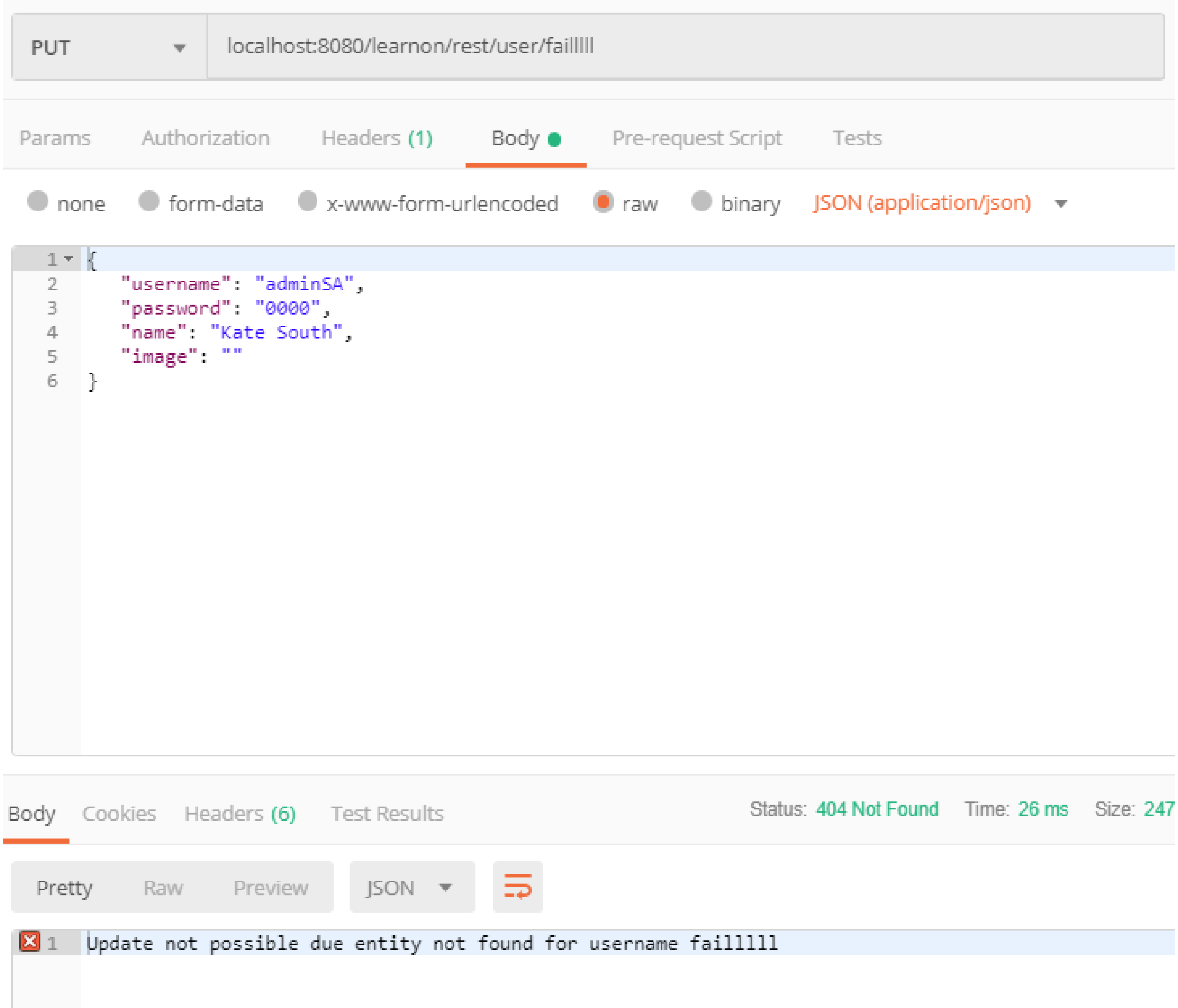
Status: 400 Bad Request Time: 10 ms Size: 252

JSON

```
Entity username (adminSA) not match with path username (AnneFiBa)
```

USER STORY 9 SCENARIO 3	As an API Consumer, I want to update a user, so that I can update an entry through my client application.
TEST CASE ID	24
DESCRIPTION	Update data based on inexistent username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/user/{inexistent username} Press send
THEN	HTTP Response 404 NOT FOUND Update not possible due entity not found for user name {inexistent username}

TEST STATUS – PASSED



PUT localhost:8080/learnon/rest/user/failllll

Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary JSON (application/json)

```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate South",
    "image": ""
}
```

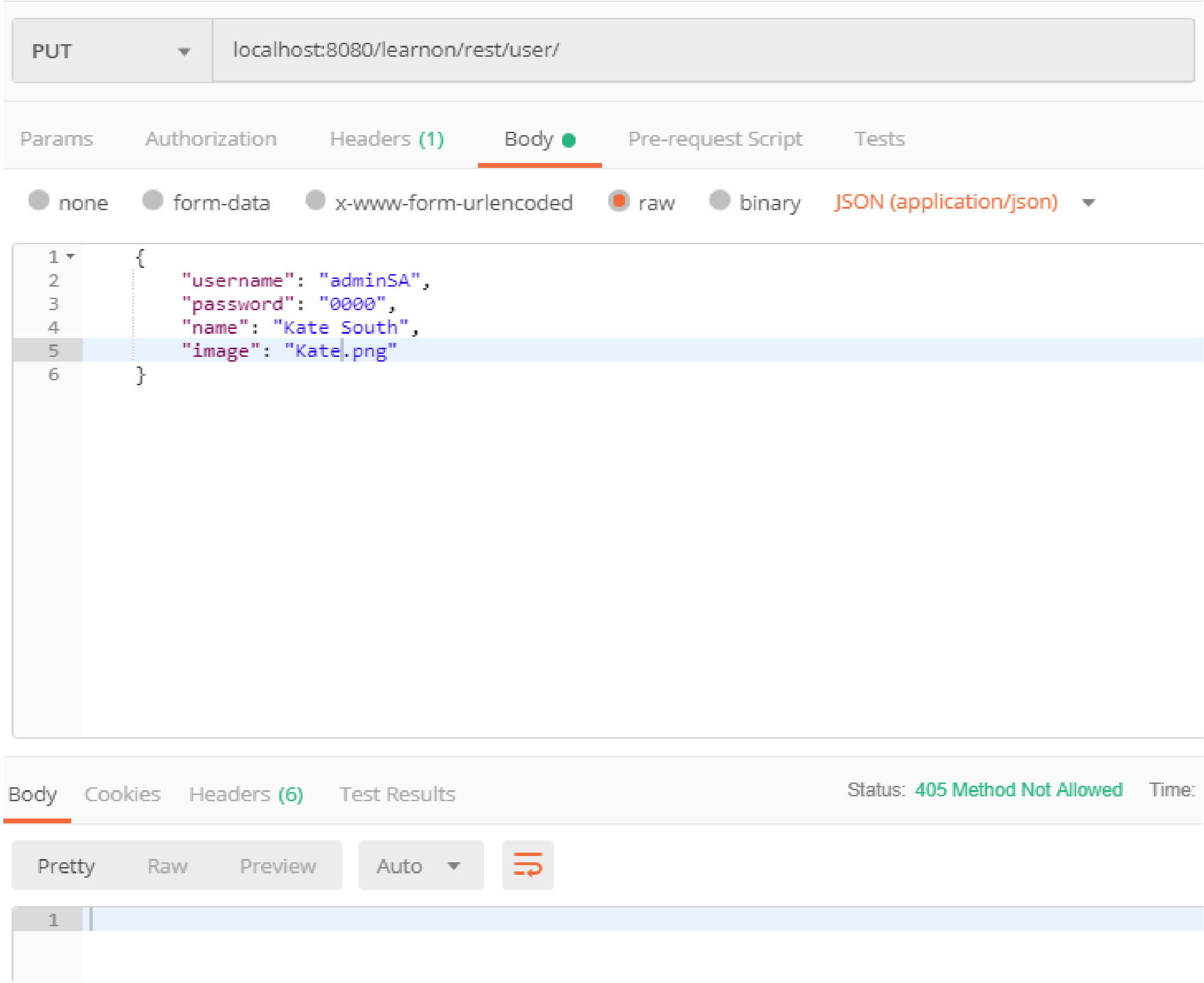
Status: 404 Not Found Time: 26 ms Size: 247

Body: JSON

```
Update not possible due entity not found for username failllll
```

USER STORY 9 SCENARIO 4	As an API Consumer, I want to update a user, so that I can update an entry through my client application.
TEST CASE ID	25
DESCRIPTION	Calling method PUT without passing username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set PUT request Enter URL http://localhost:8080/learon/rest/user/ Press send
THEN	HTTP Response 405 METHOD NOT ALLOWED User details are not updated

TEST STATUS – PASSED



PUT localhost:8080/learnon/rest/user/

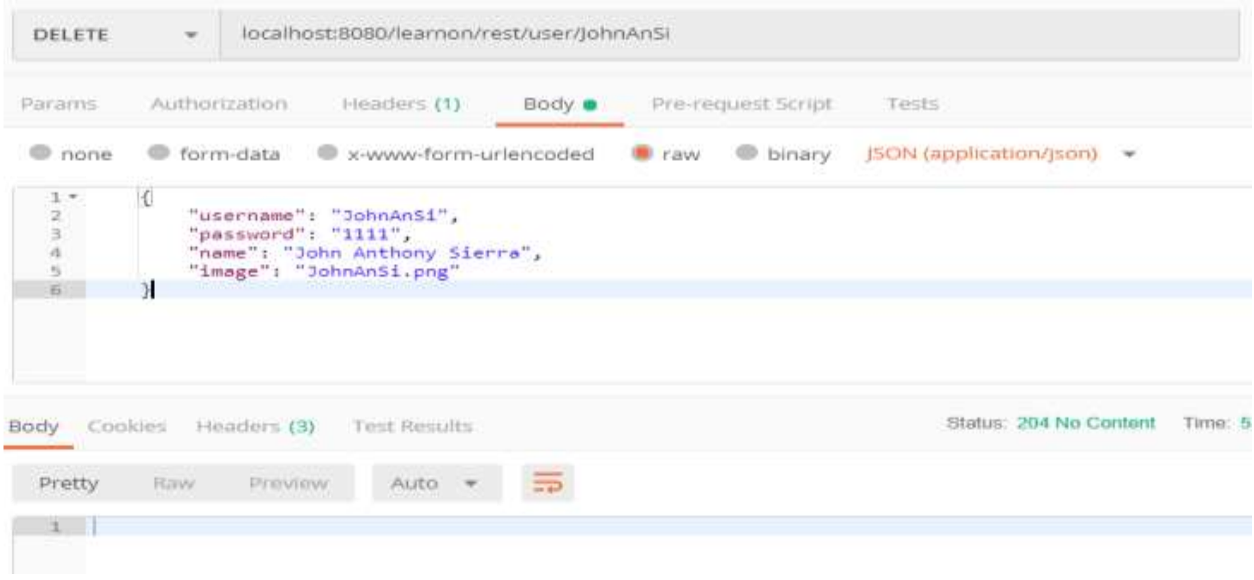
Body: raw, JSON (application/json)

```
{
    "username": "adminSA",
    "password": "0000",
    "name": "Kate South",
    "image": "Kate.png"
}
```

Status: 405 Method Not Allowed

USER STORY 10 SCENARIO 1	As an API Consumer, I want to delete a user, so that I can remove unwanted entries through my client application.
TEST CASE ID	26
DESCRIPTION	Delete data based on username and confirm that is not possible to access to a deleted data
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set DELETE request Enter URL http://localhost:8080/learon/rest/user/{valid username} Press send
THEN	HTTP Response 204 NO CONTENT The data is deleted successful so no content is showed
WHEN	Set GET request Enter URL http://localhost:8080/learon/rest/username/{deleted username} Press send
THEN	HTTP Response 200 OK User to the corresponding entered username is erased, so results an empty JSON data to the GET to the same username

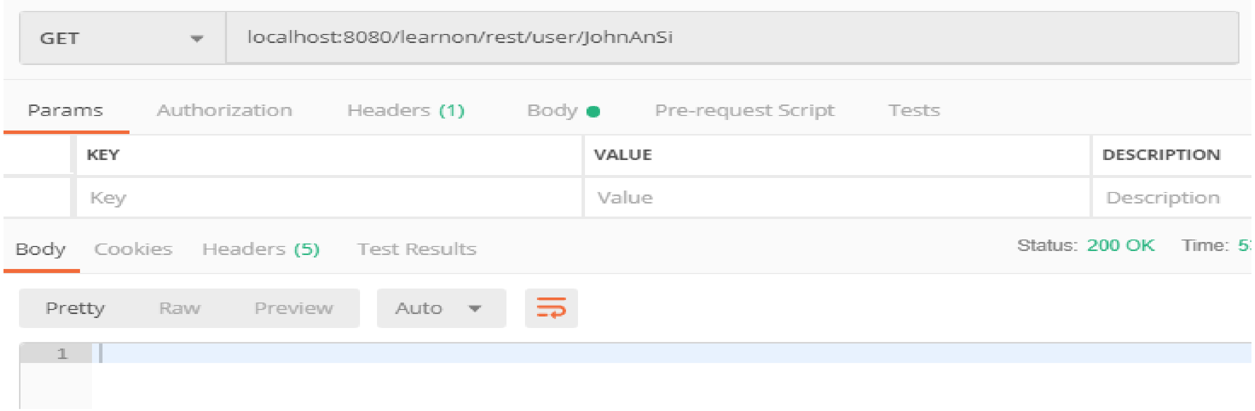
TEST STATUS – PASSED



DELETE localhost:8080/learnon/rest/user/JohnAnSi

```
{
    "username": "JohnAnSi",
    "password": "1111",
    "name": "John Anthony Sierra",
    "image": "JohnAnSi.png"
}
```

Status: 204 No Content Time: 5



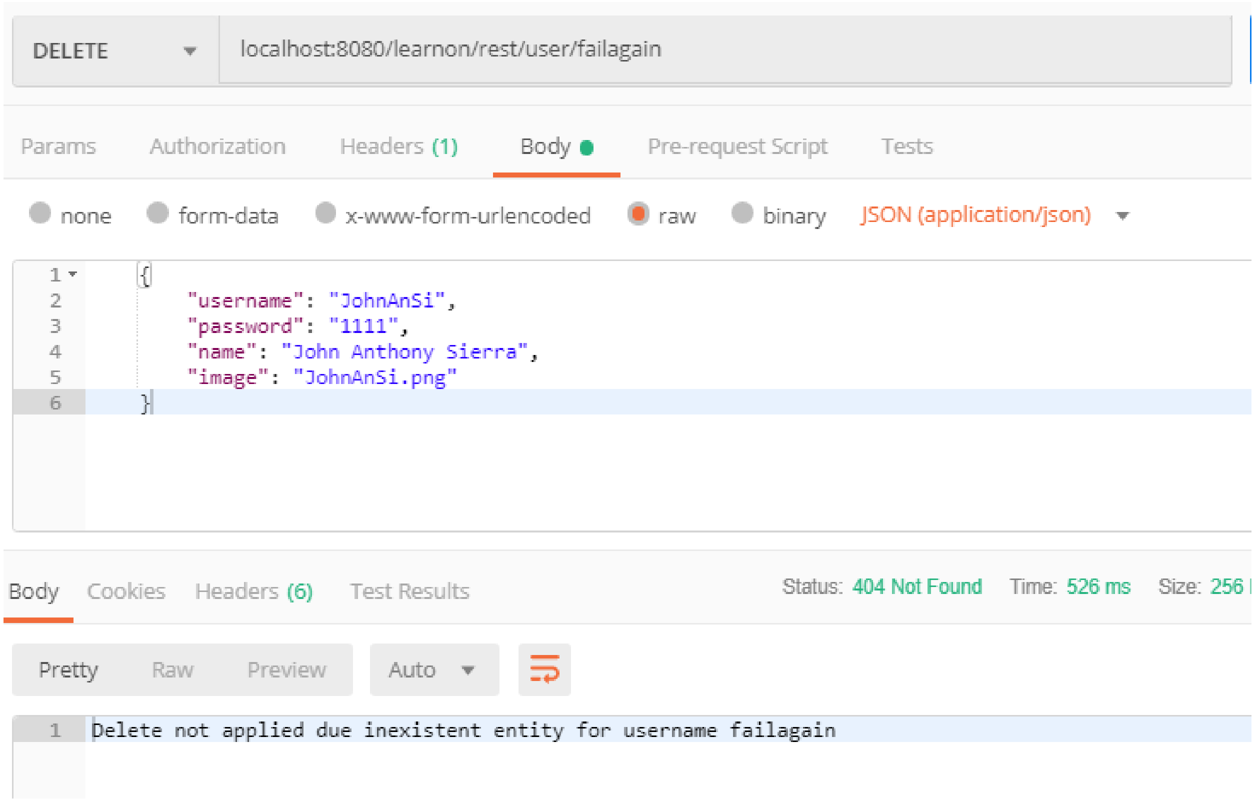
GET localhost:8080/learnon/rest/user/JohnAnSi

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 200 OK Time: 5

USER STORY 10 SCENARIO 2	As an API Consumer, I want to delete a user, so that I can remove unwanted entries through my client application.
TEST CASE ID	27
DESCRIPTION	Inexistent username is entered when deleted a user
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set DELETE request Enter URL http://localhost:8080/learon/rest/user/{inexistent username} Press send
THEN	HTTP Response 404 NOT FOUND Delete not applied due inexistent entity for username {inexistent username}

TEST STATUS – PASSED



DELETE localhost:8080/learnon/rest/user/failagain

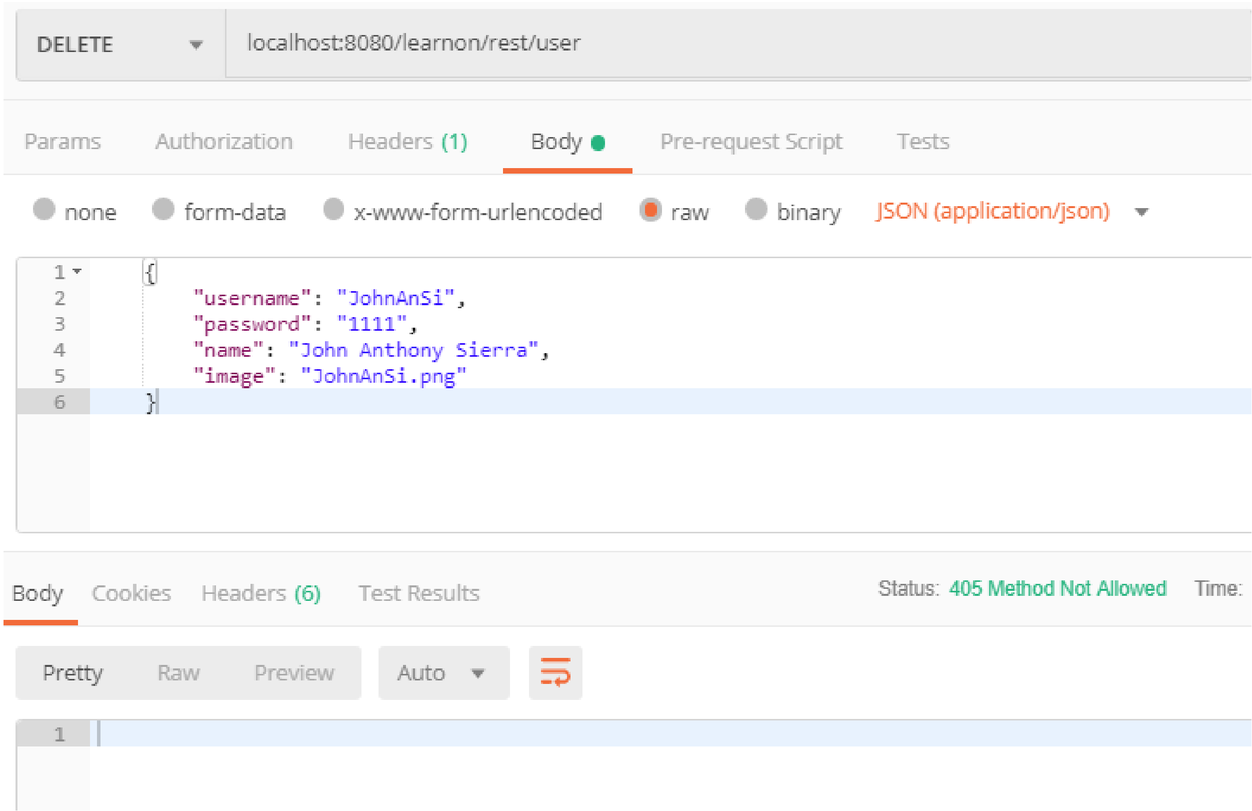
```
{
    "username": "JohnAnSi",
    "password": "1111",
    "name": "John Anthony Sierra",
    "image": "JohnAnSi.png"
}
```

Status: 404 Not Found Time: 526 ms Size: 256

```
Delete not applied due inexistent entity for username failagain
```

USER STORY 10 SCENARIO 3	As an API Consumer, I want to delete a user, so that I can remove unwanted entries through my client application.
TEST CASE ID	28
DESCRIPTION	Calling method DELETE without passing username
GIVEN	The Application Server and Database Server is running Postman REST Client Opened
WHEN	Set DELETE request Enter URL http://localhost:8080/learon/rest/user Press send
THEN	HTTP Response 405 METHOD NOT ALLOWED User is not deleted

TEST STATUS – PASSED



DELETE localhost:8080/learnon/rest/user

Body: raw JSON (application/json)

```
{
    "username": "JohnAnSi",
    "password": "1111",
    "name": "John Anthony Sierra",
    "image": "JohnAnSi.png"
}
```

Status: 405 Method Not Allowed Time:

4. TEST REPORT SUMMARY

TEST TYPE	TOTAL SCENARIOS	NUMBER PASSED	NUMBER FAILED
Manual	28	28	0

5. BUG REPORT

BUG ID	DESCRIPTION OF BUG	STATUS
US4 SC3 BI1	When trying update using an inexistent ID in the path, a new entity is created and an ID is generated.	SOLVED

There are currently no known existing bugs in this project