

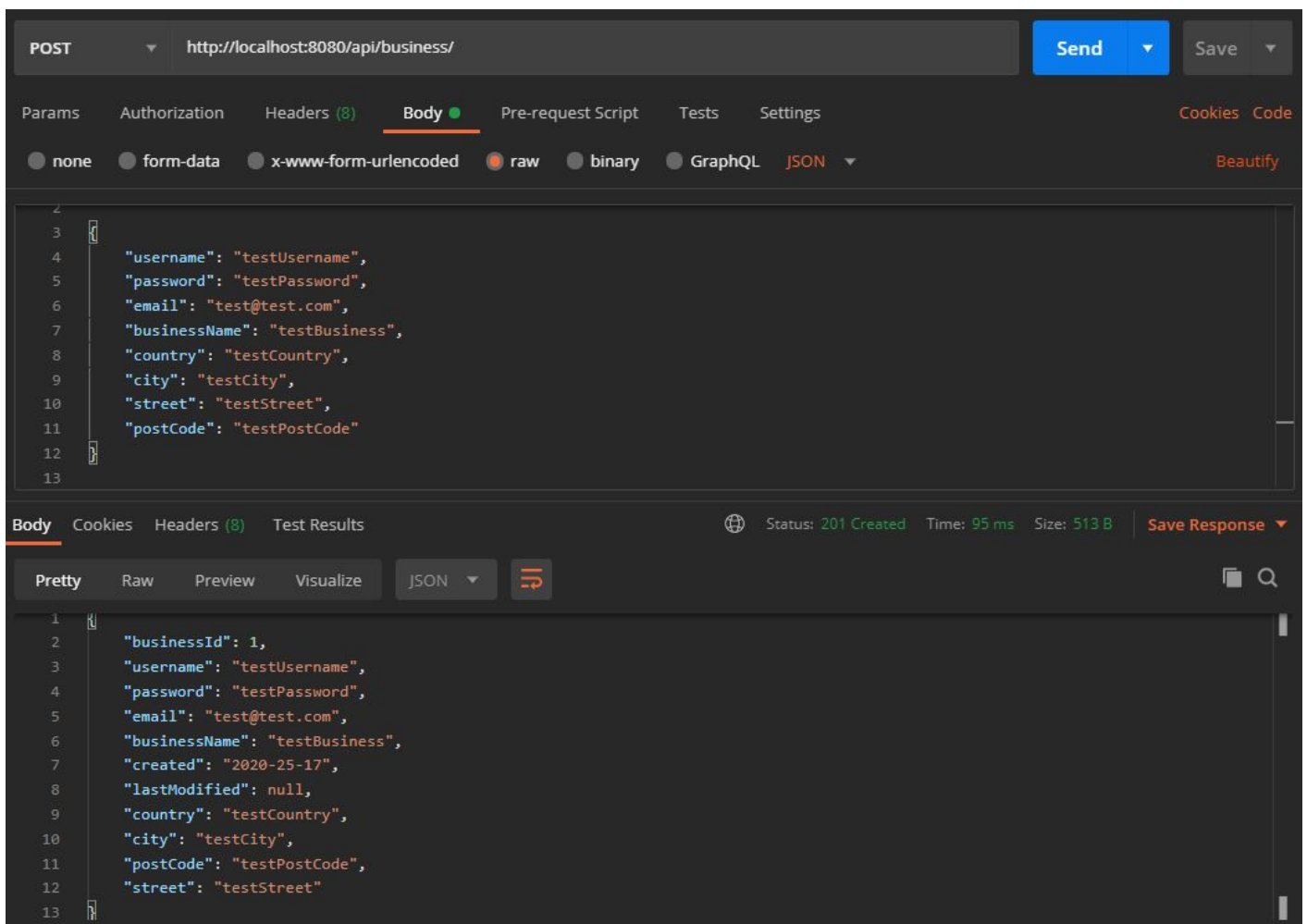
Test Case Screenshots - TUES 18:30-5

Test Case ID: 1

Test Case Description: Submit a post request for business to Backend via Postman

Test Scenario: Check Create Business Functionality

Postman:



H2 database:

The screenshot shows the H2 Console interface. The left sidebar displays the database schema with tables: ADDRESS, BUSINESS, PERSON, STUDENT, USER, and INFORMATION_SCHEMA. The main area shows the SQL statement: `SELECT * FROM BUSINESS;`. Below the statement, the results are displayed in a table with 7 columns: BUSINESS_ID, BUSINESS_NAME, CREATED, EMAIL, LAST_MODIFIED, PASSWORD, and USERNAME. The first row contains the values: 1, testBusiness, 2020-09-17 21:25:36.838, test@test.com, null, testPassword, and testUsername. The execution time is 3 ms.

SQL statement:

```
SELECT * FROM BUSINESS;
```

BUSINESS_ID	BUSINESS_NAME	CREATED	EMAIL	LAST_MODIFIED	PASSWORD	USERNAME
1	testBusiness	2020-09-17 21:25:36.838	test@test.com	null	testPassword	testUsername

(1 row, 3 ms)

Edit

The screenshot shows the H2 Console interface. The left sidebar displays the database schema with tables: ADDRESS, BUSINESS, PERSON, STUDENT, USER, and INFORMATION_SCHEMA. The main area shows the SQL statement: `SELECT * FROM ADDRESS;`. Below the statement, the results are displayed in a table with 5 columns: CITY, COUNTRY, POST_CODE, STREET, and BUSINESS_ID. The first row contains the values: testCity, testCountry, testPostCode, testStreet, and 1. The execution time is 2 ms.

SQL statement:

```
SELECT * FROM ADDRESS;
```

CITY	COUNTRY	POST_CODE	STREET	BUSINESS_ID
testCity	testCountry	testPostCode	testStreet	1

(1 row, 2 ms)

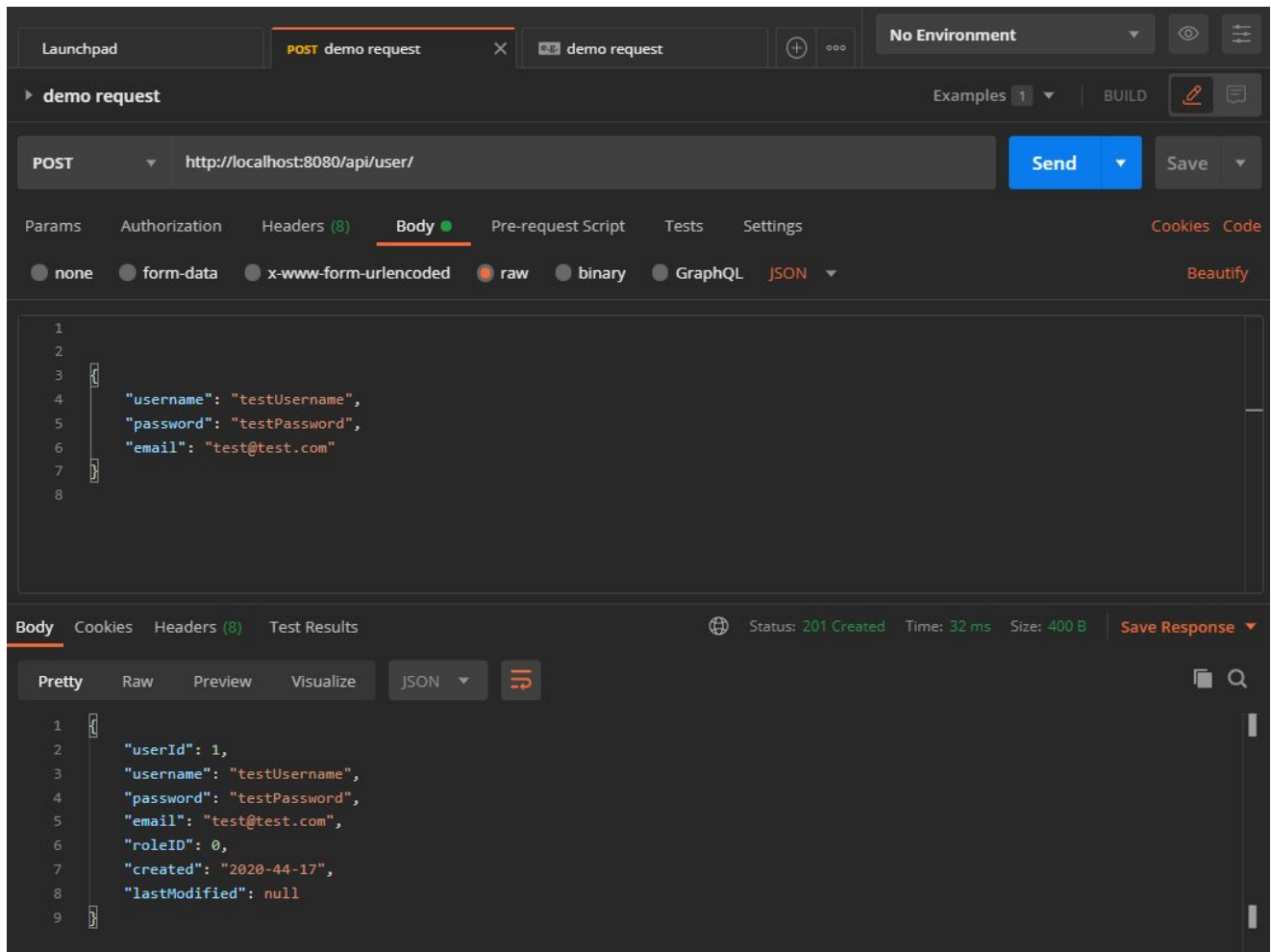
Edit

Test Case ID: 2

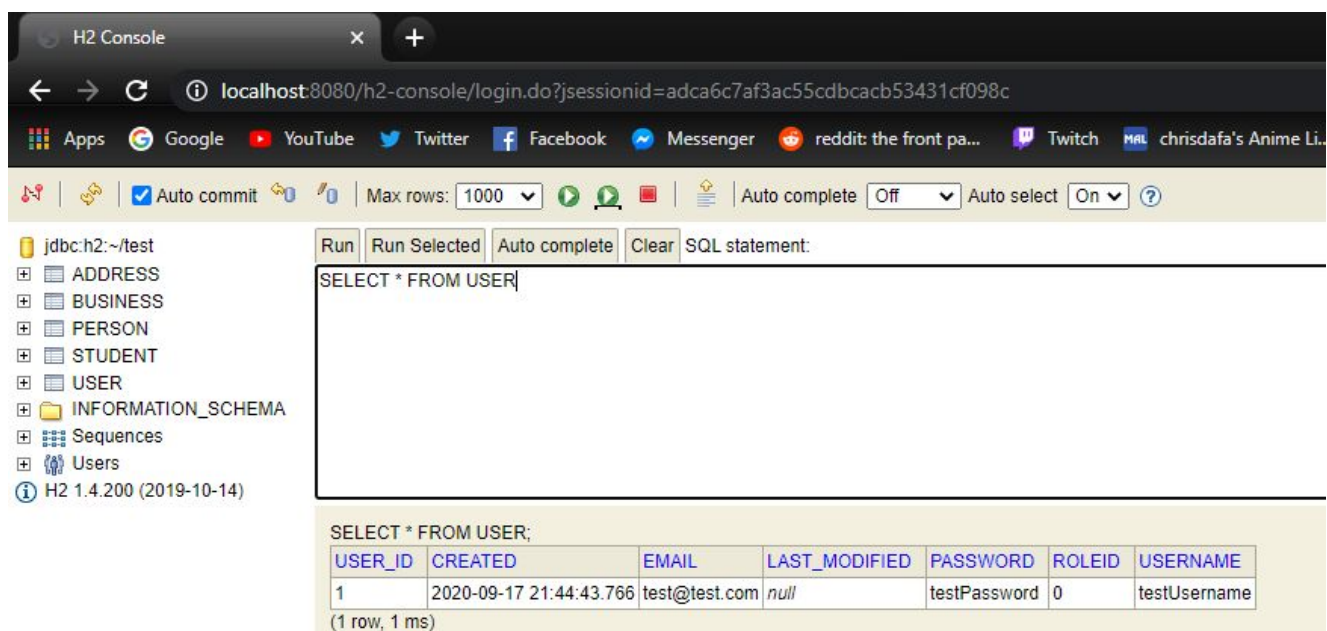
Test Case Description: Submit a post request for user to Backend via Postman

Test Scenario: Check Create User Functionality

Postman:



H2 database:

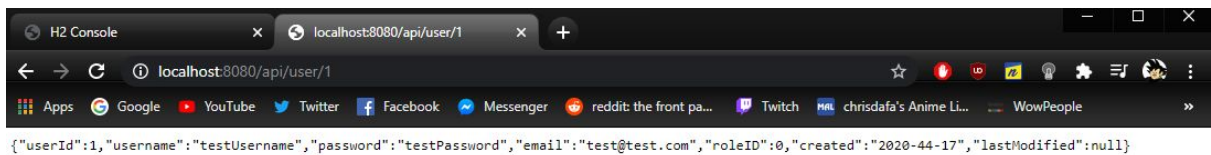


Test Case ID: 3

Test Case Description: send a get request for user to Backend via Postman

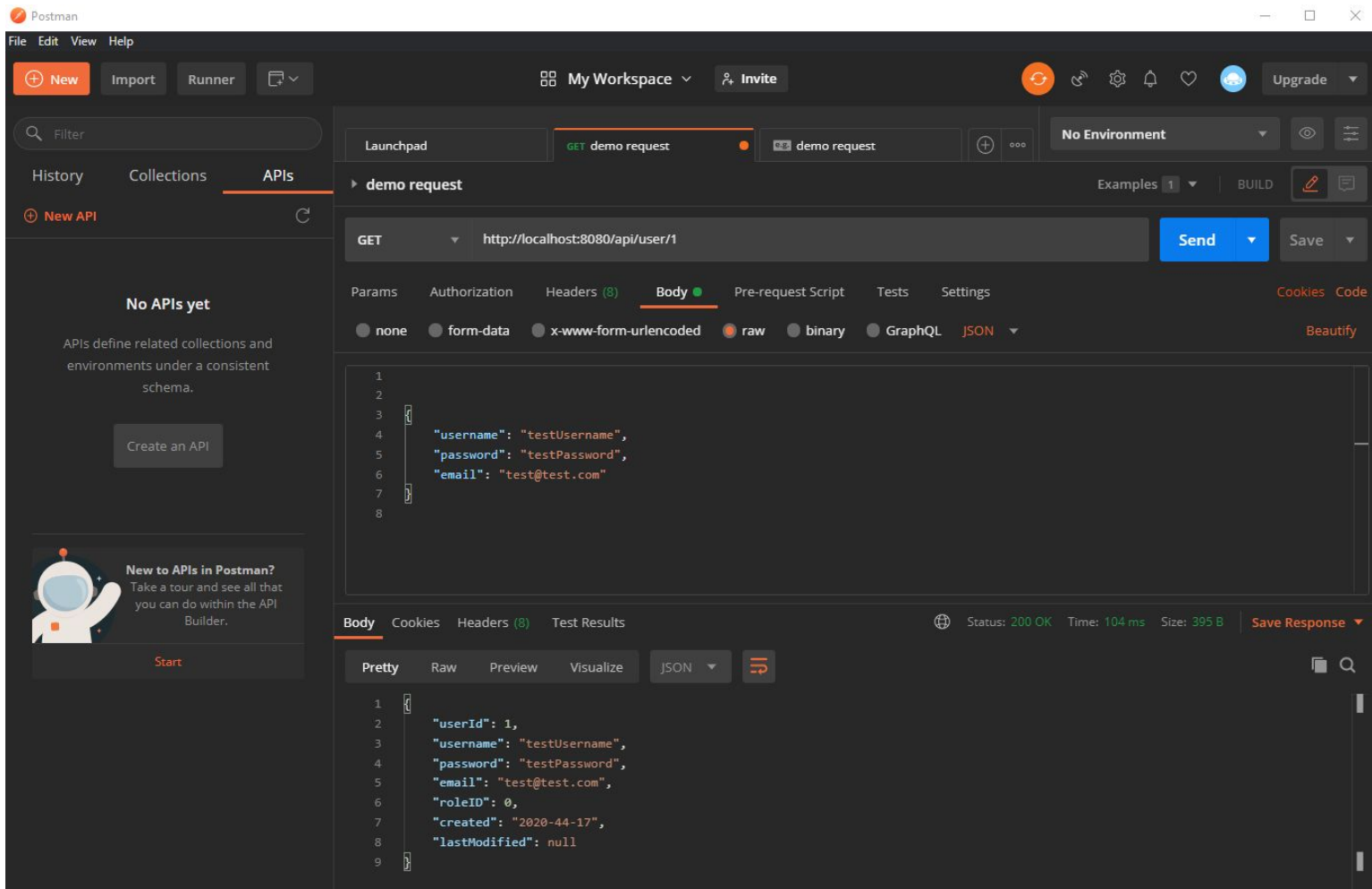
Test Scenario: Check Get User Functionality

User details returned:



```
{ "userId": 1, "username": "testUsername", "password": "testPassword", "email": "test@test.com", "roleID": 0, "created": "2020-44-17", "lastModified": null }
```

Postman:

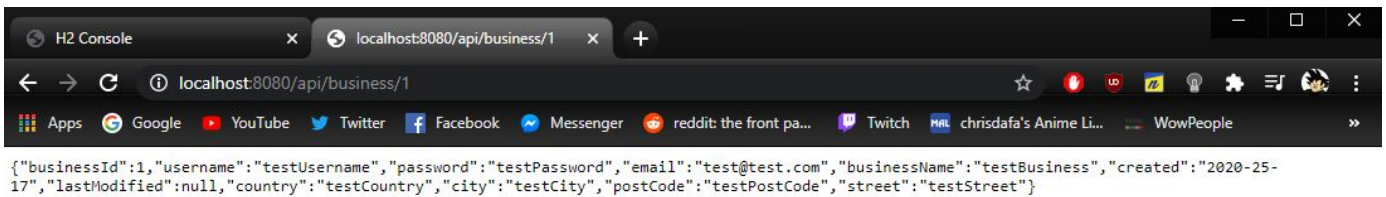


Test Case ID: 4

Test Case Description: send a get request for business to Backend via Postman

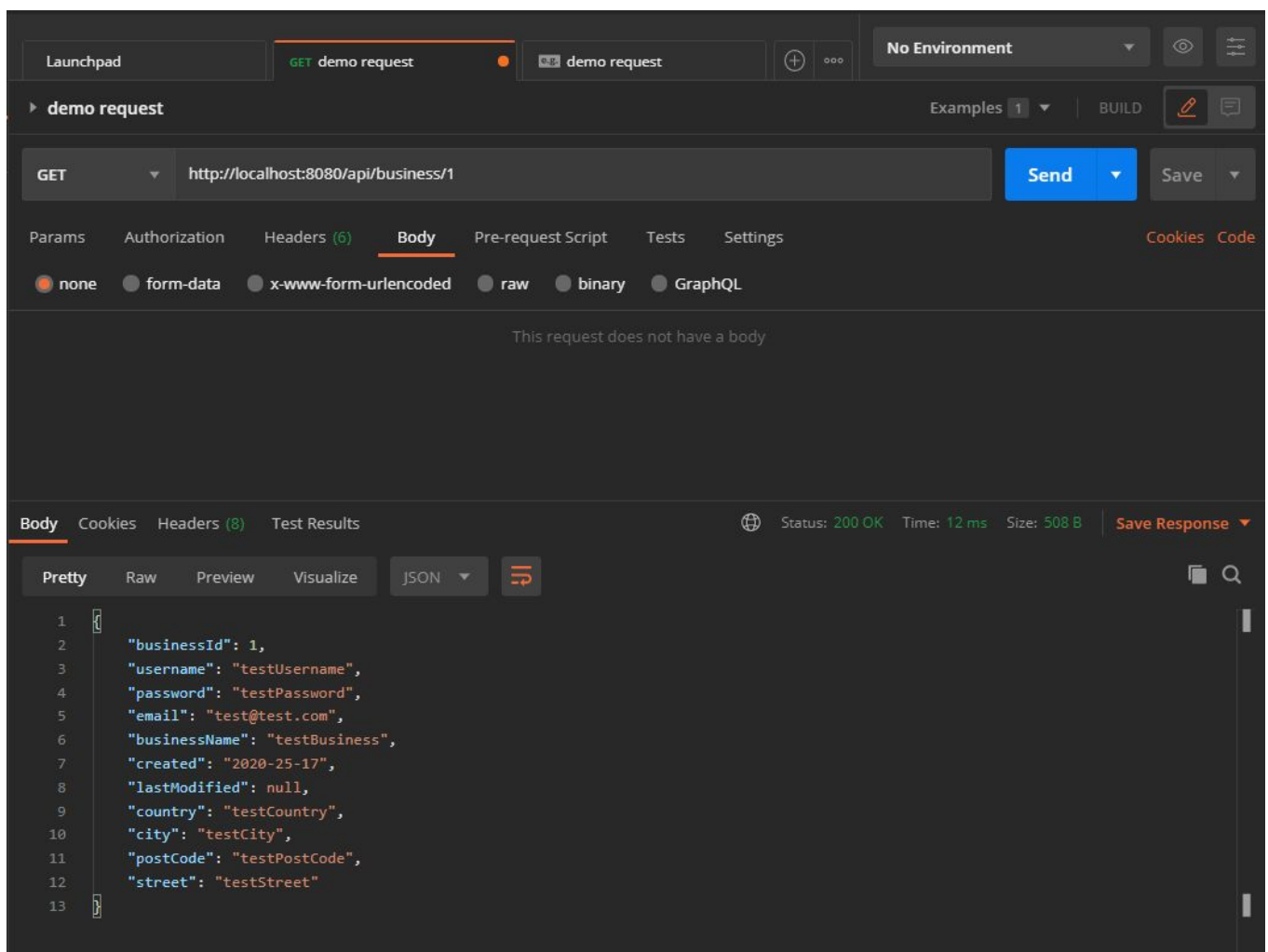
Test Scenario: Check Get Business Functionality

Business details returned:



```
{
  "businessId": 1,
  "username": "testUsername",
  "password": "testPassword",
  "email": "test@test.com",
  "businessName": "testBusiness",
  "created": "2020-25-17",
  "lastModified": null,
  "country": "testCountry",
  "city": "testCity",
  "postCode": "testPostCode",
  "street": "testStreet"
}
```

Postman:

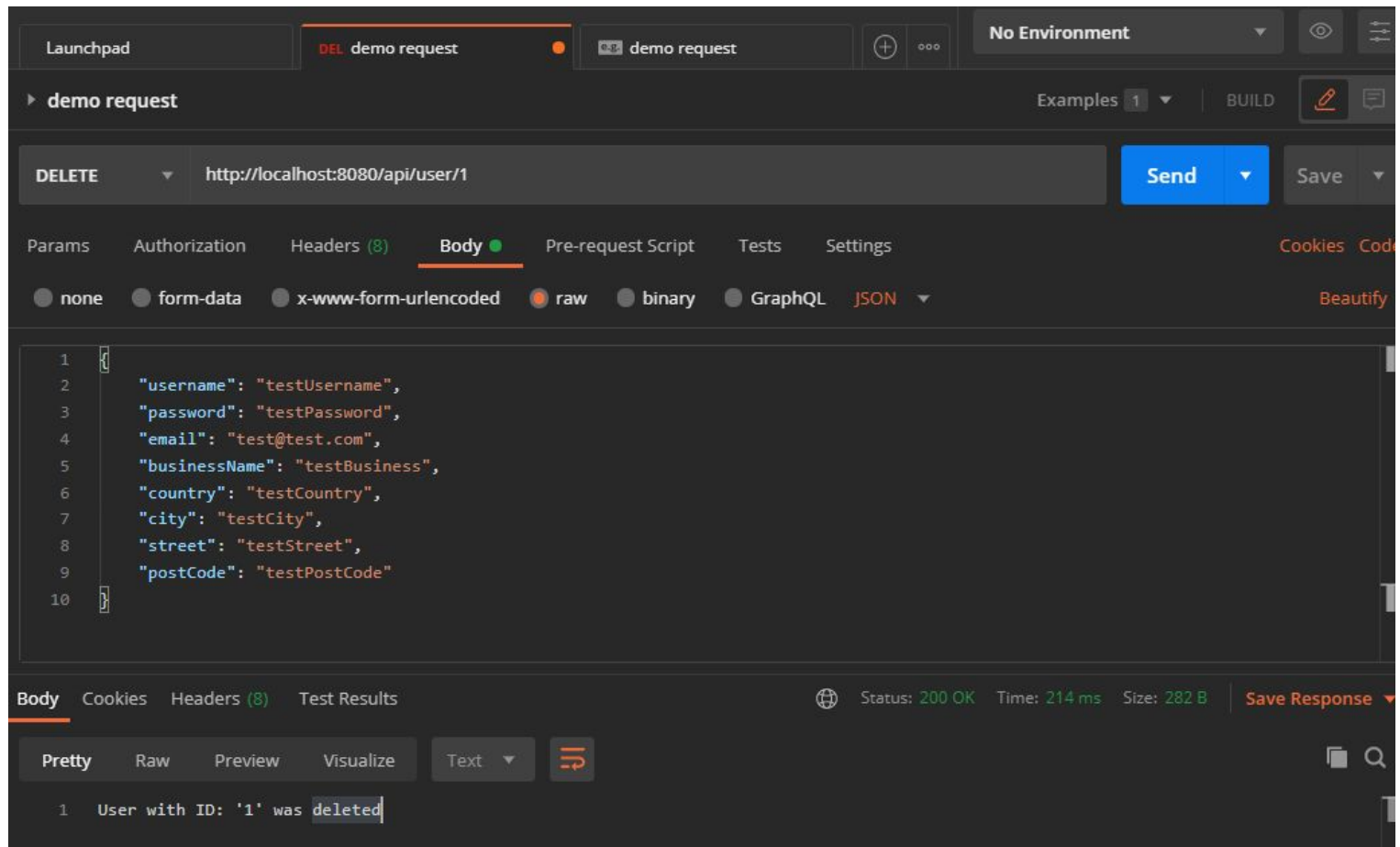


Test Case ID: 5

Test Case Description: send a delete request for user to Backend via Postman

Test Scenario: Check Delete User Functionality

Postman:

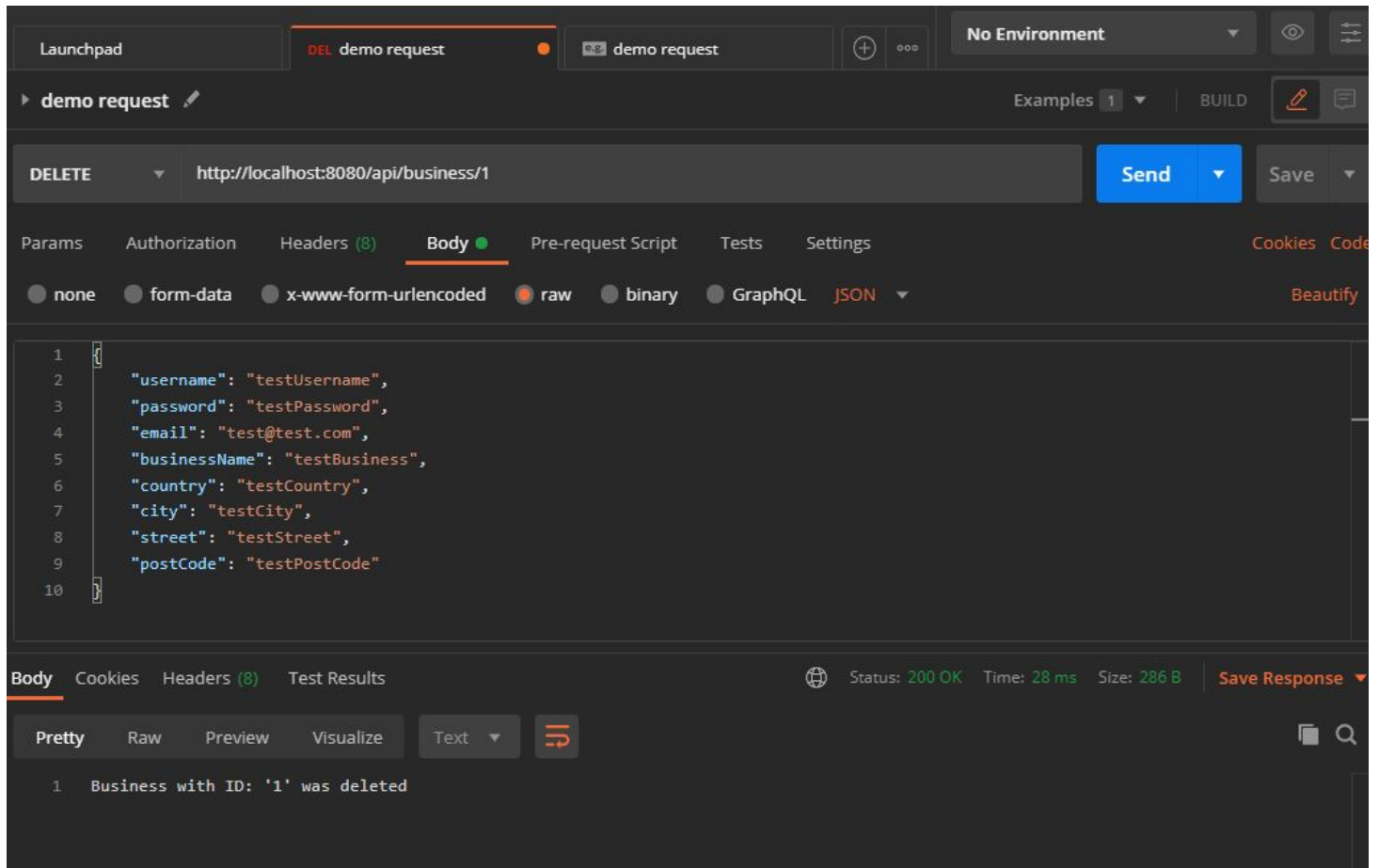


Test Case ID: 6

Test Case Description: send a delete request for business to Backend via Postman

Test Scenario: Check Delete Business Functionality

Postman:

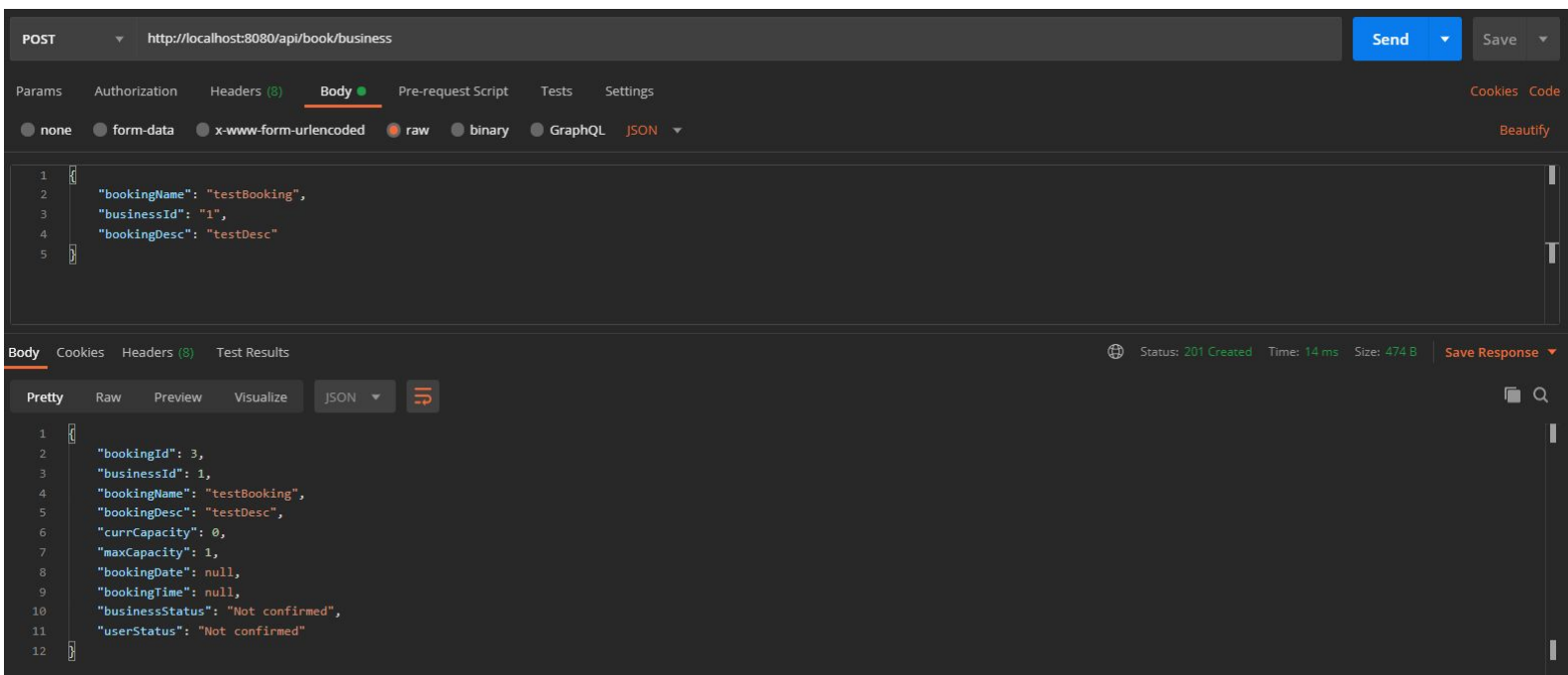


Test Case ID: 7

Test Case Description: send a post request for business booking to Backend via Postman

Test Scenario: Check Create Business Booking Functionality

Postman:



Database:

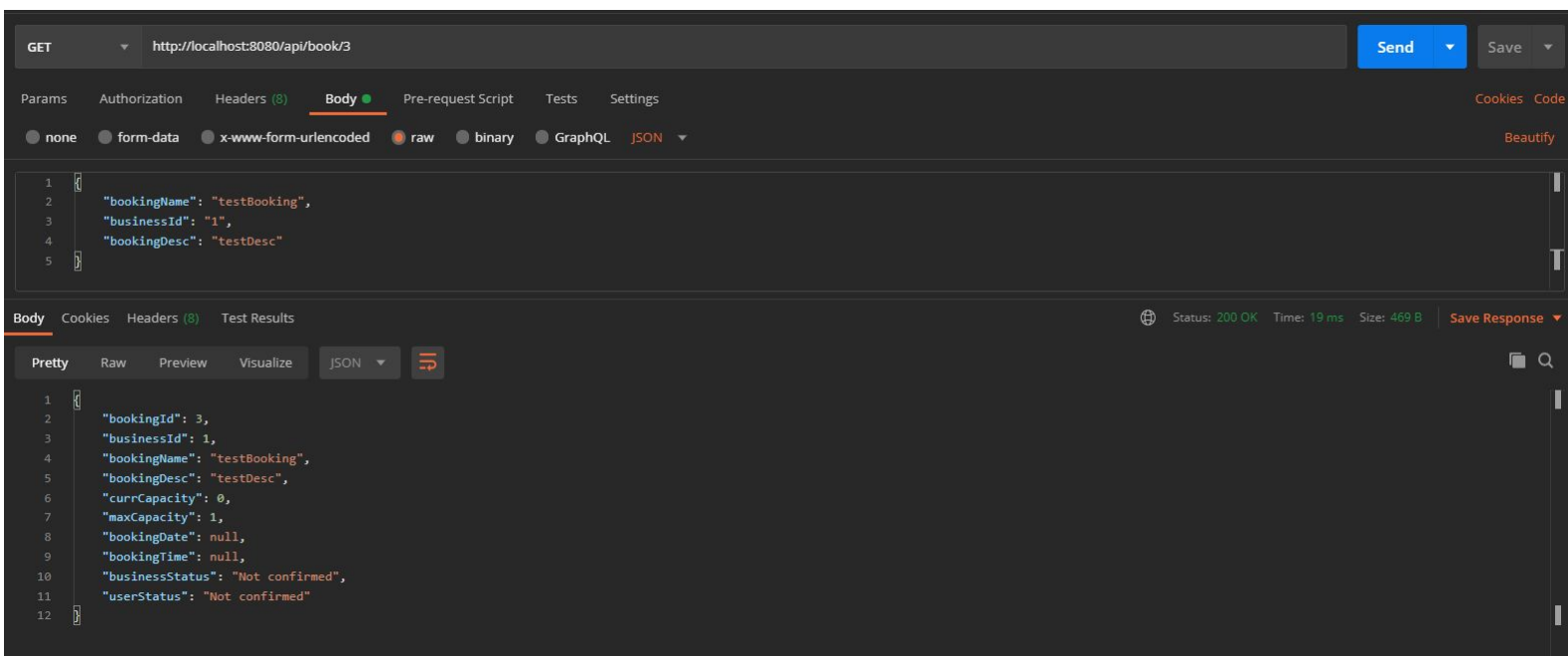


Test Case ID: 8

Test Case Description: send a get request for booking via booking ID to Backend via Postman

Test Scenario: Check Get Booking via Booking ID Functionality

Postman:

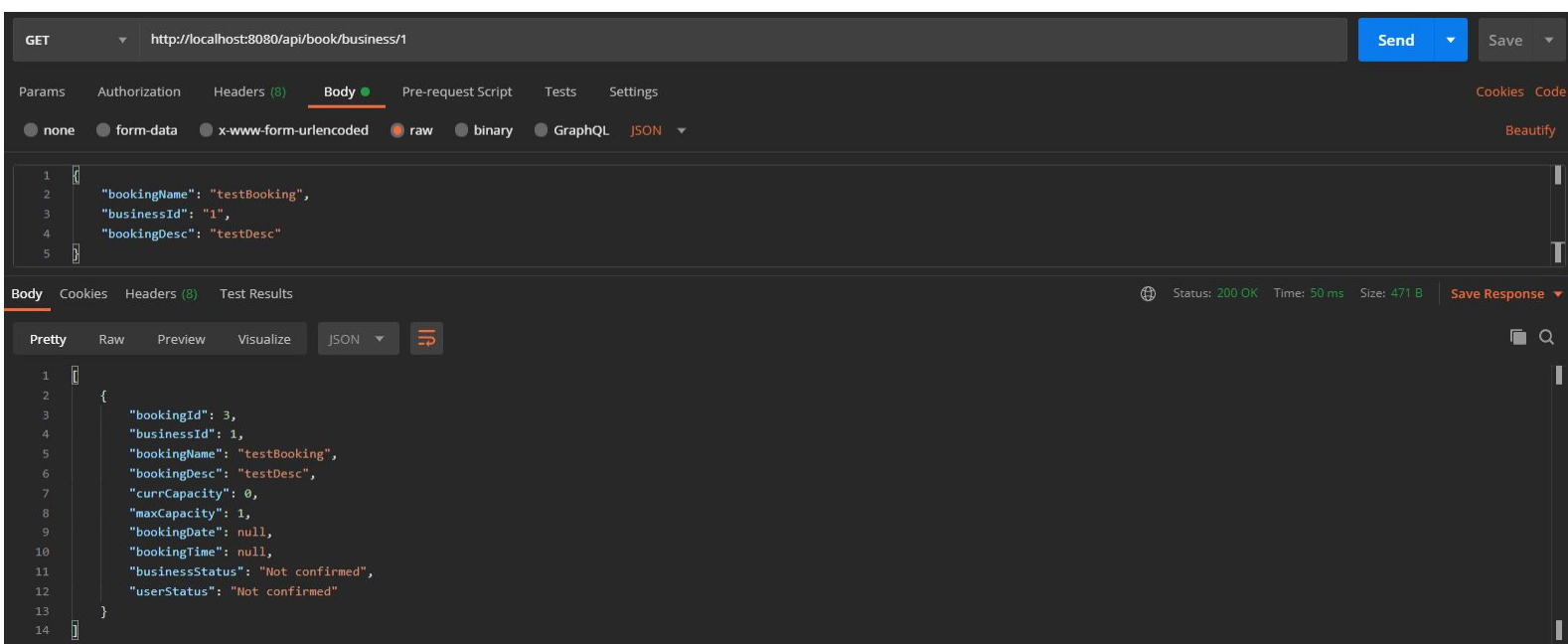


Test Case ID: 9

Test Case Description: send a get request for booking via business ID to Backend via Postman

Test Scenario: Check Get Booking via business ID Functionality

Postman:



Test Case ID: 10

Test Case Description: Submit a post request with missing postCode for business to Backend via Postman

Test Scenario: Check Create Business with missing postCode Functionality

Postman:

