

Payment Gateway – API

Payment Gateway contains three APIs to process payment, retrieve payment by id and retrieve all the processed payment details.

Postman – Using postman I have tested all the API locally.

Local host port number for my application during build: **7210**. Please build the application locally and use your application port number in API URL.

Authorization: (Basic Auth)

Implemented “Basic Auth” type authorization to access all the APIs.

Username: **admin**

Password: **password**

Authorization tab – Selected “Basic Auth” type and entered username & password.

I will work on this to get & check valid user from DB in future.

Status: 401 Unauthorized authentications

The screenshot shows the Postman interface for a POST request to `https://localhost:7210/api/payment/`. The request is configured with Basic Authentication, with the username `admin` and password `password1`. The response status is `401 Unauthorized`, with a response time of `236 ms` and a body size of `102 B`. The interface includes tabs for Params, Authorization, Headers (9), Body, Pre-request Script, Tests, Settings, and Cookies. A warning message states: "Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#)".

POST `https://localhost:7210`

`https://localhost:7210/api/payment/`

POST `https://localhost:7210/api/payment/`

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

Type Basic A...
The authorization header will be automatically generated when you send the request. Learn more about [authorization](#)

Username `admin`

Password `password1`

Body Cookies Headers (3) Test Results

401 Unauthorized 236 ms 102 B Save as Example

List of APIs:

1. Process Payment – POST method

URL: <https://localhost:7210/api/payment/>

Method: Post

Body: Payment JSON object

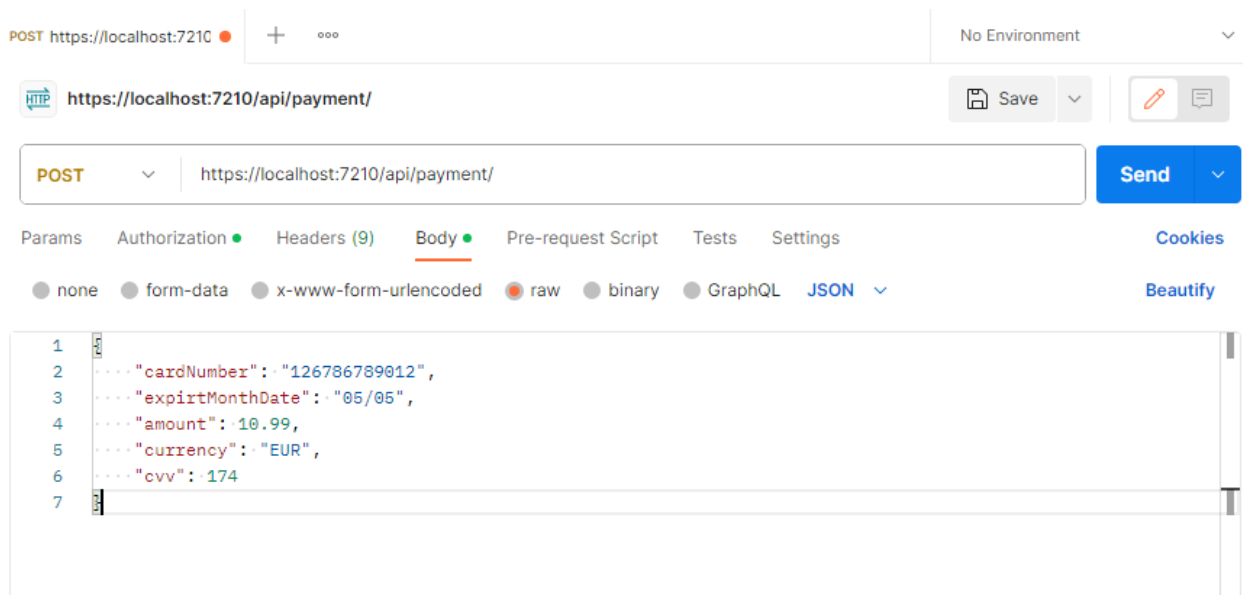
Payment Id will automatically generate so no need to include that in the object.

JSON object:

```
{  
  "cardNumber": "126786789012",  
  "expirtMonthDate": "05/05",  
  "amount": 10.99,  
  "currency": "EUR",  
  "cvv": 174  
}
```

- All fields are required.
- Validations:
 - Card number should be 12 characters.
 - Currency should be 3 characters.
 - CVV should be 3 digits range between 000-999.

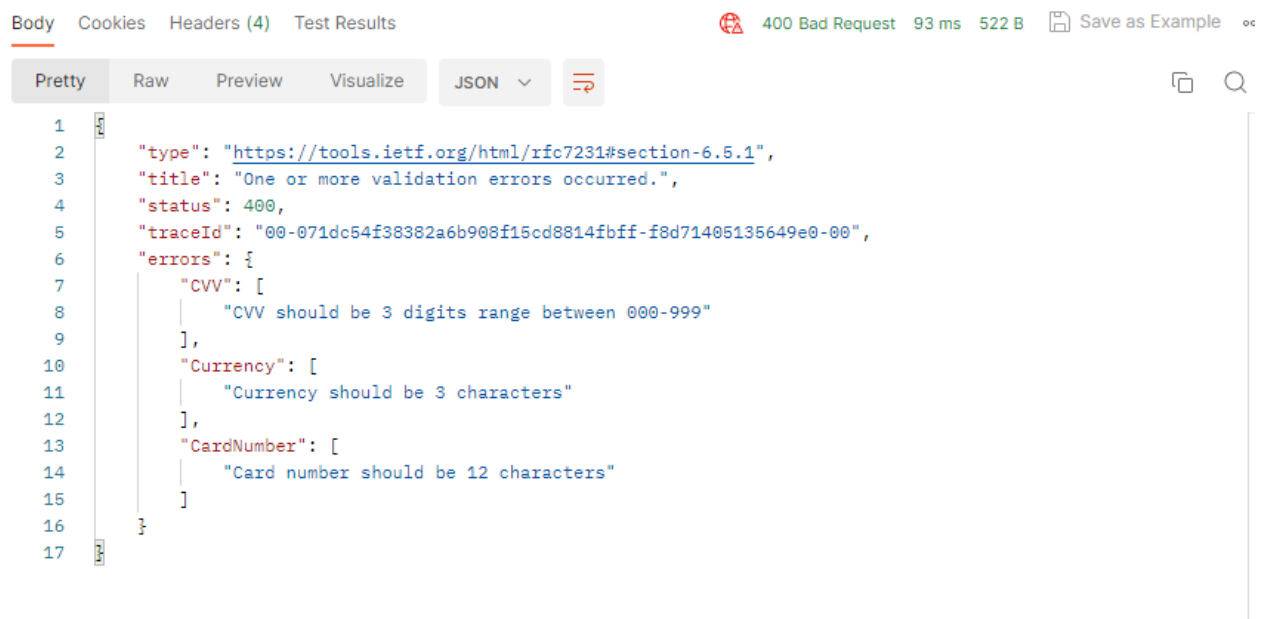
We should pass the Json object in request body.



Status: 200 OK (payment successful)



Status: 400 Bad Request (payment Unsuccessful)



2. Retrieve Payment by Id – GET method:

URL: <https://localhost:7210/api/payment/{paymentid}>

Method: Get

Example: <https://localhost:7210/api/payment/d1b7411f-1bad-411d-ae66-faf1904b6f93>

Mask Card Number – Retrieve masked card number. Only the last 4 characters are visible.

Request:

GET https://localhost:7210/ + ... No Environment

https://localhost:7210/api/payment/d1b7411f-1bad-411d-ae66-faf1904b6f93 Save

GET https://localhost:7210/api/payment/d1b7411f-1bad-411d-ae66-faf1904b6f93 Send

Params Authorization Headers (7) **Body** Pre-request Script Tests Settings Cookies

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

This request does not have a body

Response: (valid payment id) – Status 200 OK

Body Cookies Headers (4) Test Results 200 OK 373 ms 289 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "d1b7411f-1bad-411d-ae66-faf1904b6f93",
3   "cardNumber": "*****9012",
4   "expirtMonthDate": "05/05",
5   "amount": 10.99,
6   "currency": "EUR",
7   "cvv": 174
8 }
```

Response: (Invalid payment id) – Status 204 No content

Body Cookies Headers (3) Test Results 204 No Content 48 ms 100 B Save as Example

Pretty Raw Preview Visualize Text

1

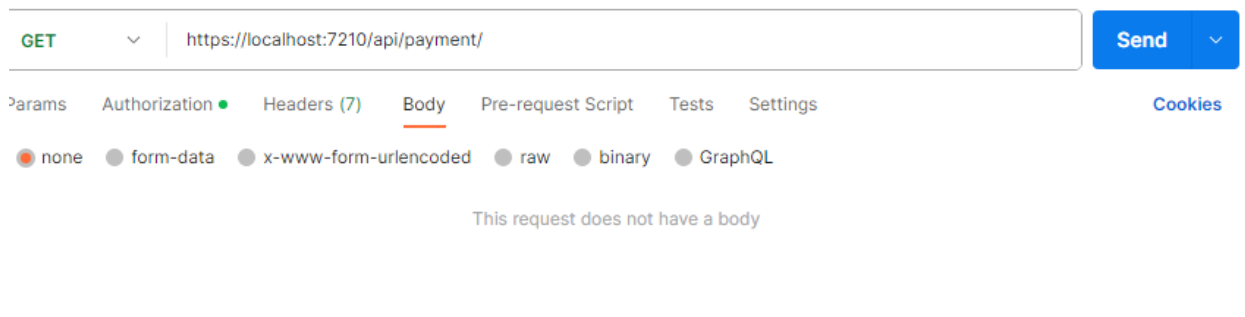
3. Retrieve all the Processed payments– GET method:

URL: <https://localhost:7210/api/payment/>

Method: Get

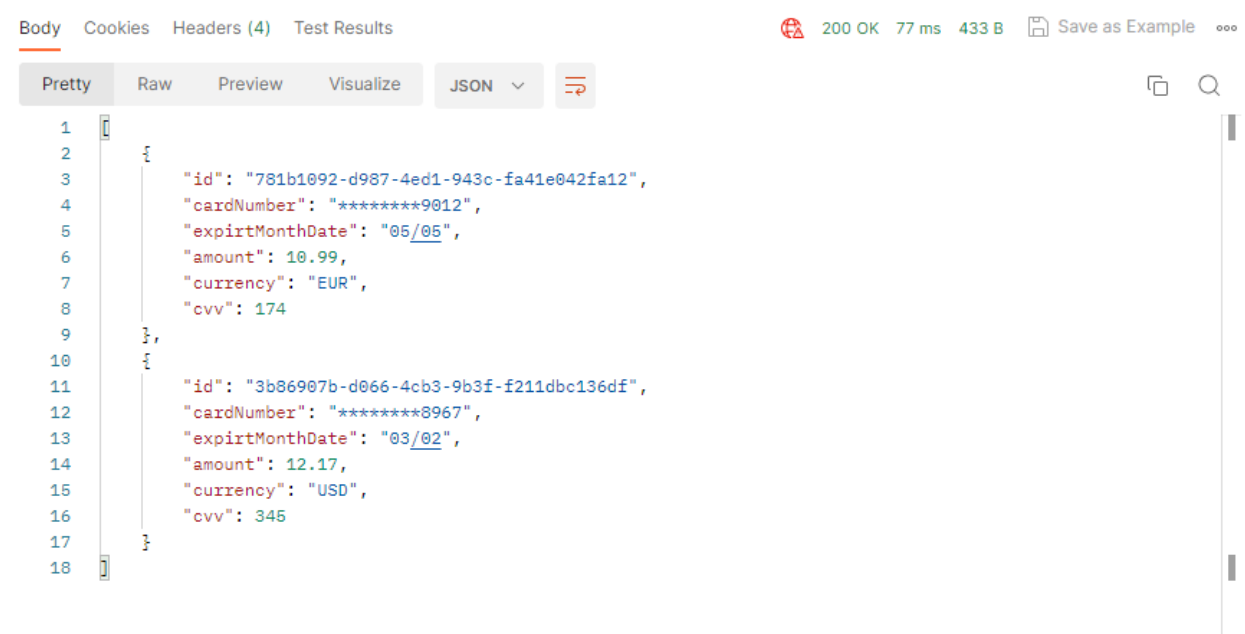
This API is helpful to find the all the processed payments.

Request:



The screenshot shows a REST client interface with a GET request to <https://localhost:7210/api/payment/>. The interface includes tabs for Params, Authorization, Headers (7), Body, Pre-request Script, Tests, and Settings. The Body tab is selected, and it shows a message: "This request does not have a body". There are also radio buttons for different content types: none, form-data, x-www-form-urlencoded, raw, binary, and GraphQL.

Response:



The screenshot shows the response of the GET request. The status is 200 OK, with a response time of 77 ms and a size of 433 B. The response is displayed in JSON format, showing two payment objects. The first object has an id of "781b1092-d987-4ed1-943c-fa41e042fa12", a card number of "*****9012", an expiration date of "05/05", an amount of 10.99, a currency of "EUR", and a cvv of 174. The second object has an id of "3b86907b-d066-4cb3-9b3f-f211dbc136df", a card number of "*****8967", an expiration date of "03/02", an amount of 12.17, a currency of "USD", and a cvv of 345.

```
1 {
2   {
3     "id": "781b1092-d987-4ed1-943c-fa41e042fa12",
4     "cardNumber": "*****9012",
5     "expirtMonthDate": "05/05",
6     "amount": 10.99,
7     "currency": "EUR",
8     "cvv": 174
9   },
10  {
11    "id": "3b86907b-d066-4cb3-9b3f-f211dbc136df",
12    "cardNumber": "*****8967",
13    "expirtMonthDate": "03/02",
14    "amount": 12.17,
15    "currency": "USD",
16    "cvv": 345
17  }
18 }
```