Installation

Checkout Payment Gateway

Use Command Line

1. **Navigate to the containing folder:**

```
cd {Path}\CheckoutPaymentGatewayApi\Server\CheckoutPaymentGateway
```

2. **Build the solution:**

```
docker build -f Dockerfile -t checkoutpaymentgateway ..
```

3. **Check the image has been created:**

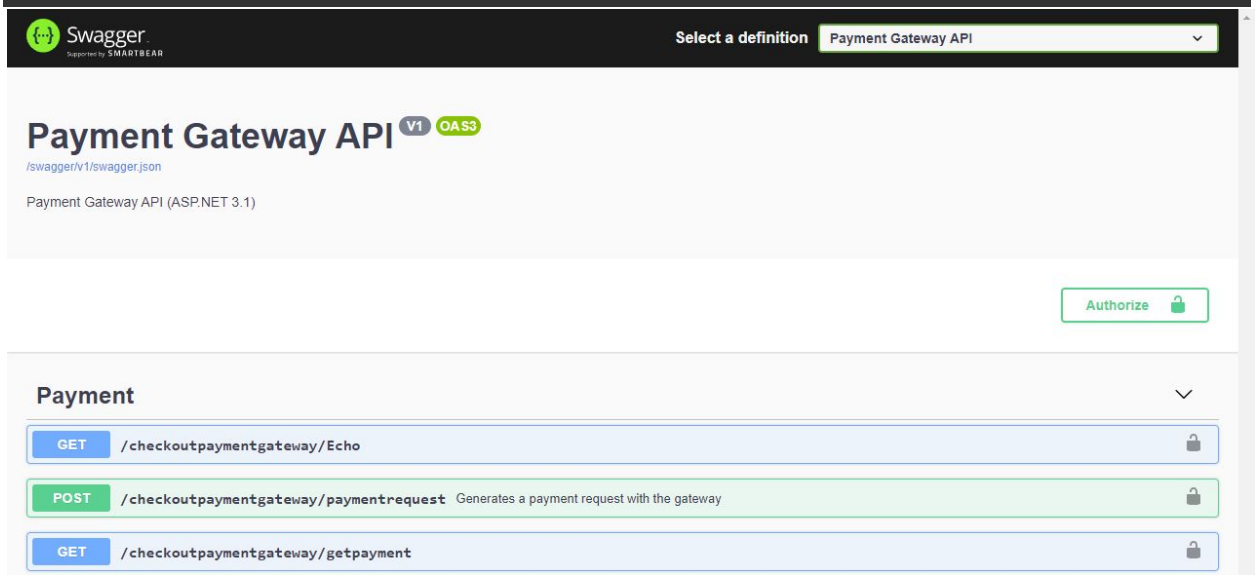
```
docker images | more
```

4. **Create the docker container:**

```
docker-compose up
```

5. **Navigate to the Api:**

```
https://localhost:6001/Swagger
```



Prometheus

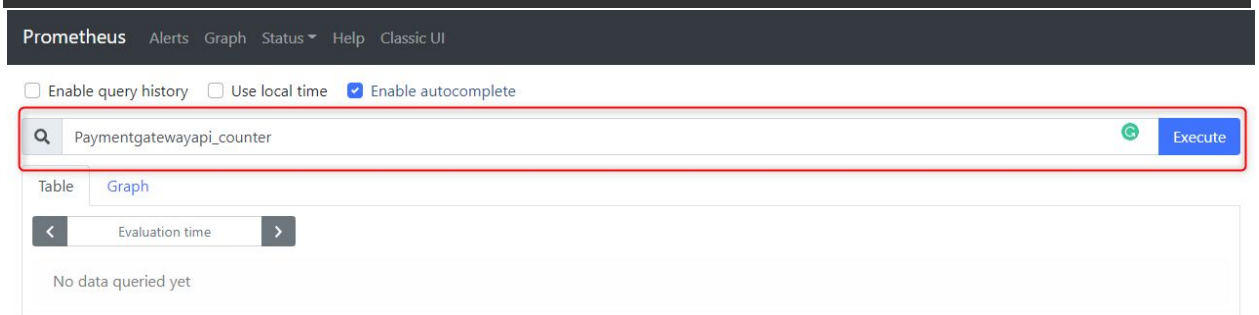
Use Command Line

1. **Navigate to Prometheus for metrics:**

```
http://localhost:9090/graph
```

2. **Select the metrics to show:**

```
Paymentgatewayapi_counter
```



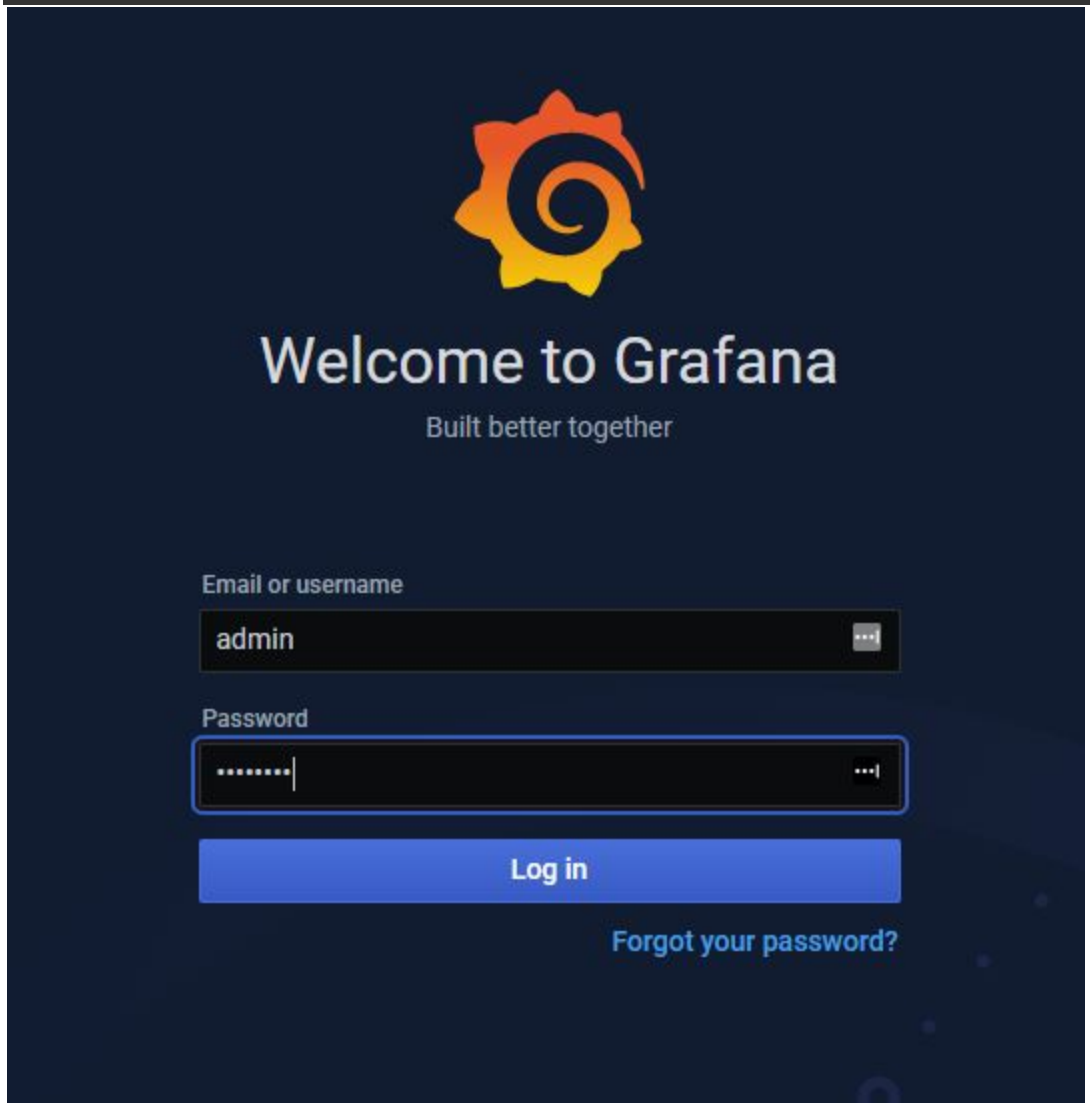
3. **Click Execute**

Grafana

Use Command Line

1. **Navigate to Grafana:**

```
http://localhost:3000/login
```



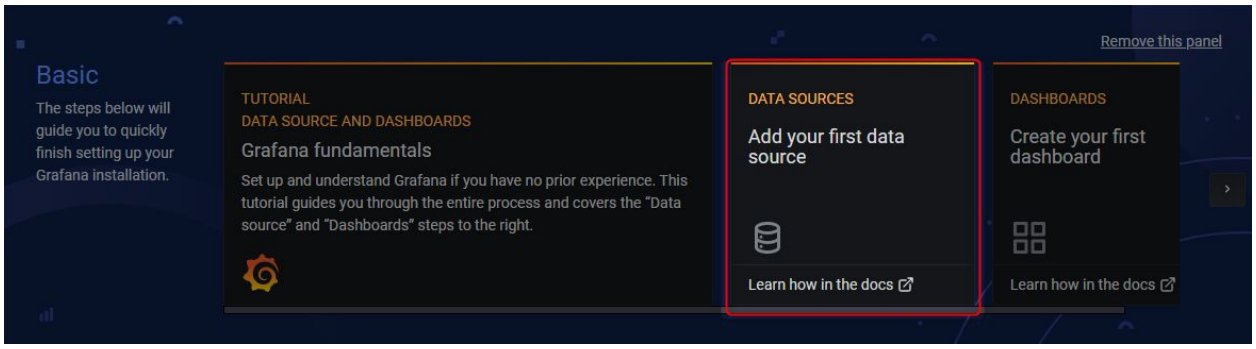
2. **Log in:**

```
username: admin
```

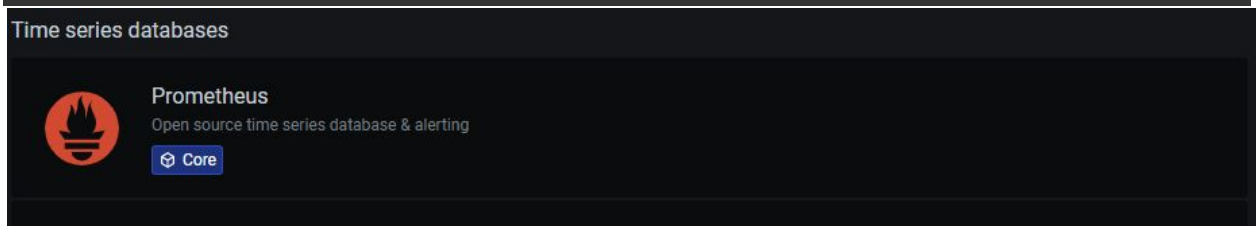
```
password: P@ssw0rd
```

3. **Add Datasource**

```
Click Add your first data sourse
```

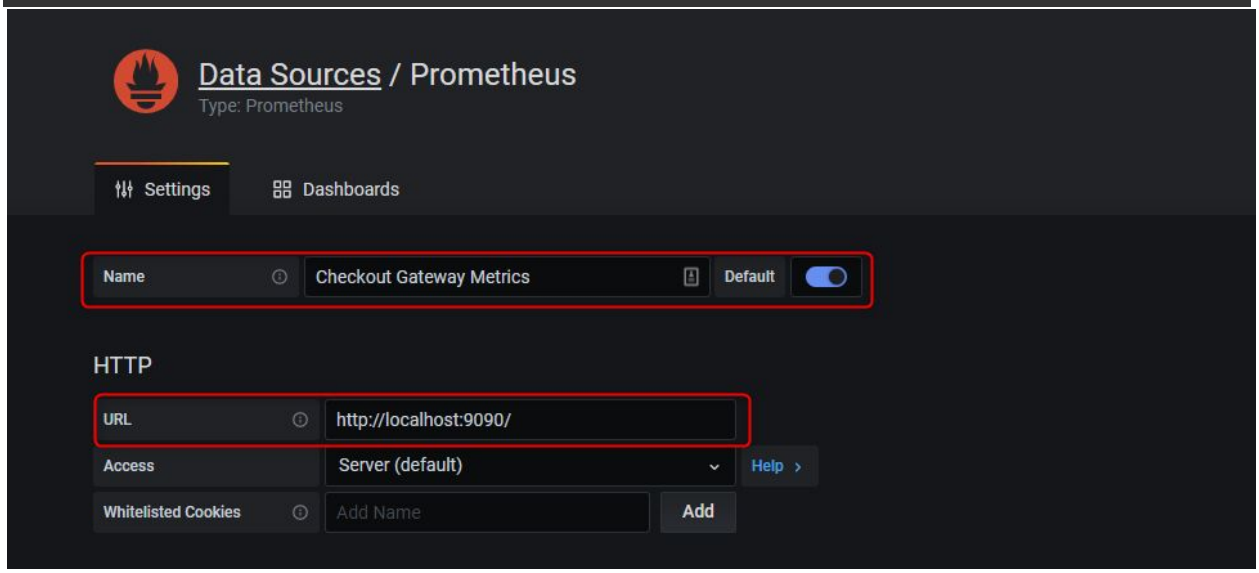



```
Click Prometheus
```

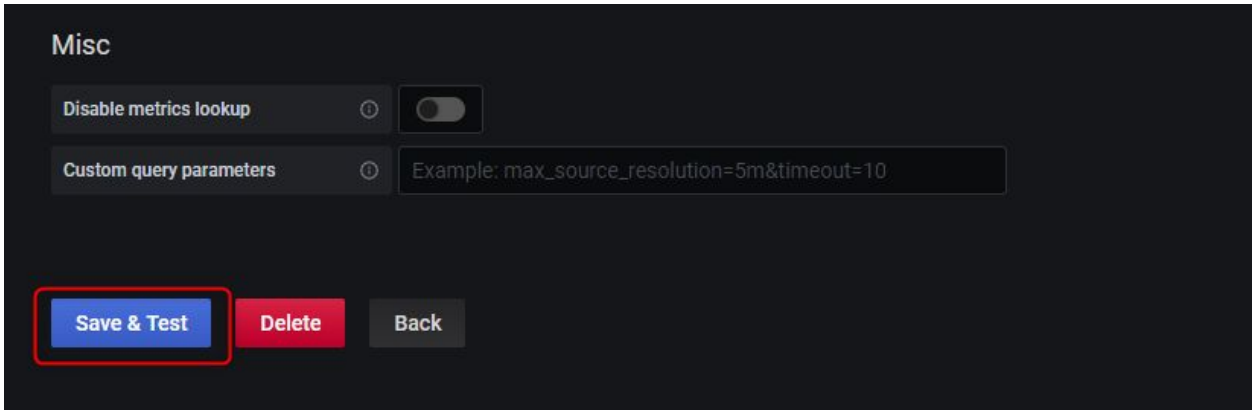



```
Create a useful name i.e. checkout gateway metrics
```

```
Add the URL: http://localhost:9090/
```

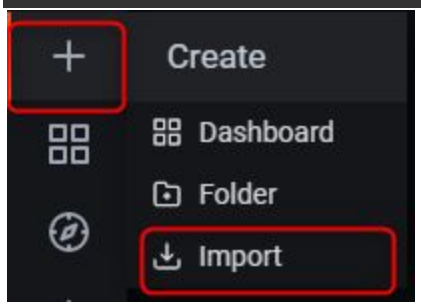



```
Click Save & Test
```

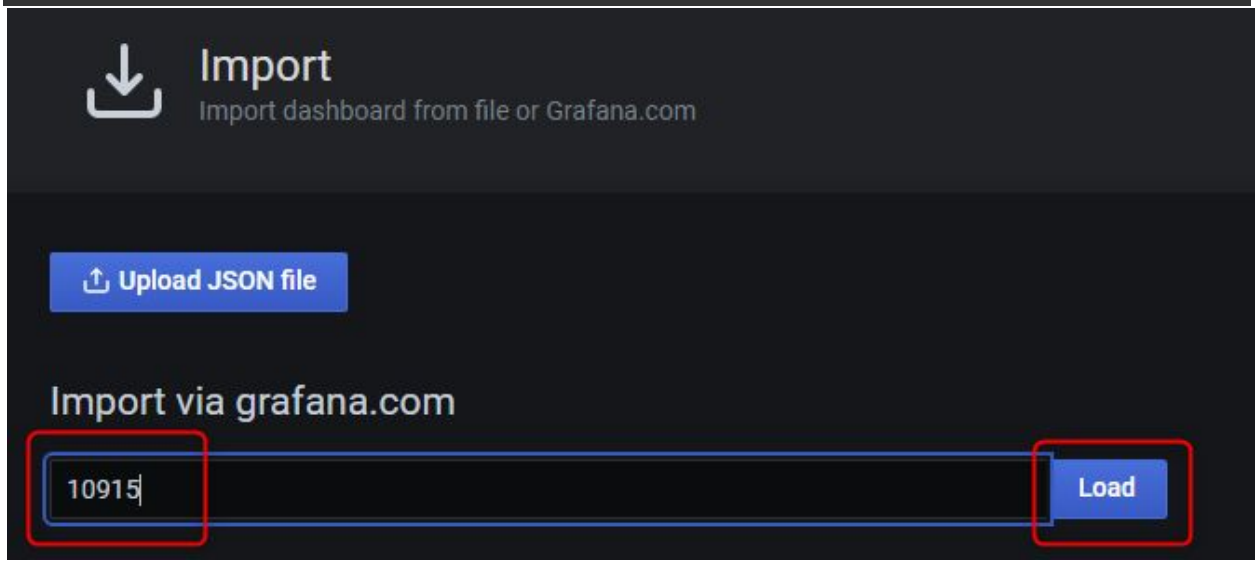


4. **Add Dashboard**
    a. **Example dashboard imports:**

```
Click the + on the left of the screen
```



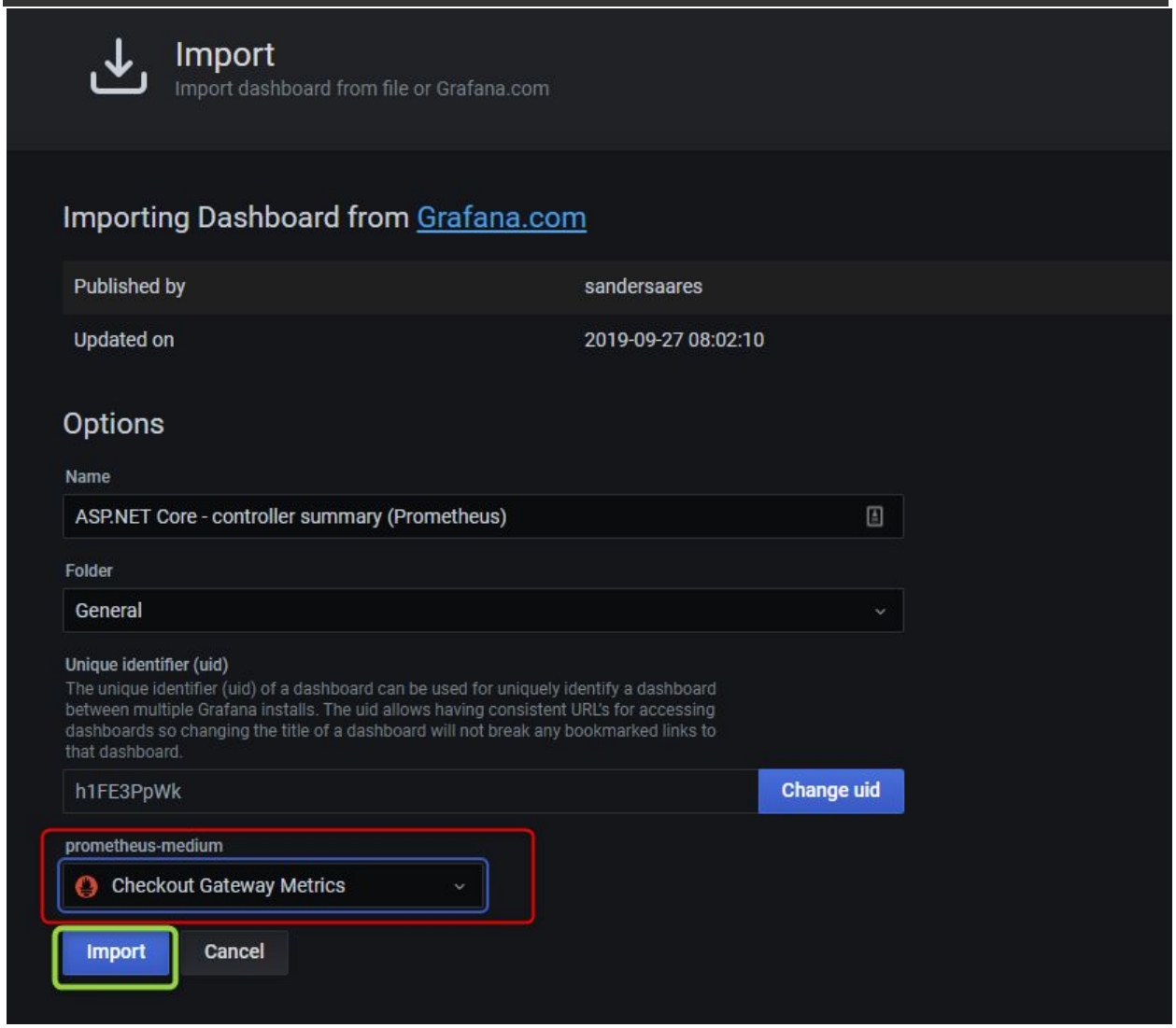

```
Type the id: 10427 or the id 10915 to load preset dashboards
```




```
Click Load
```

```
Select the data source you created from the drop-down (above import)
```

```
Click Import
```



A dashboard can also be built from scratch

SQL Database

Use PowerShell

1. **Download latest SQL server 2019:**

```
docker pull mcr.microsoft.com/mssql/server:2019-latest
```

2. **Run SQL Server linux docker container. NOTE: Password: $!DH7q94**

```
docker run -e "ACCEPT_EULA=Y" -e "SA_PASSWORD=$!DH7q94" `
   -p 1633:1433 --name PaymentTechTest -h PaymentTechTest `
   -d mcr.microsoft.com/mssql/server:2019-latest
```

Note this is a developer version of SQL Server.

3. **Change the SA PASSWORD. New Password: $!DH7q94**

```
docker exec -it PaymentTechTest /opt/mssql-tools/bin/sqlcmd `
   -S localhost -U SA -P "$!DH7q94" `
   -Q "ALTER LOGIN SA WITH PASSWORD='%zYG614&'"
```

4. **Connect to the SQL Server**

```
docker exec -it PaymentTechTest "bash"
```

5. **Connect locally to sql command:**

```
/opt/mssql-tools/bin/sqlcmd -S localhost -U SA -P "%zYG614&"
```

6. **Publish the PaymentsDb to the container**

Running

Bearer Token:

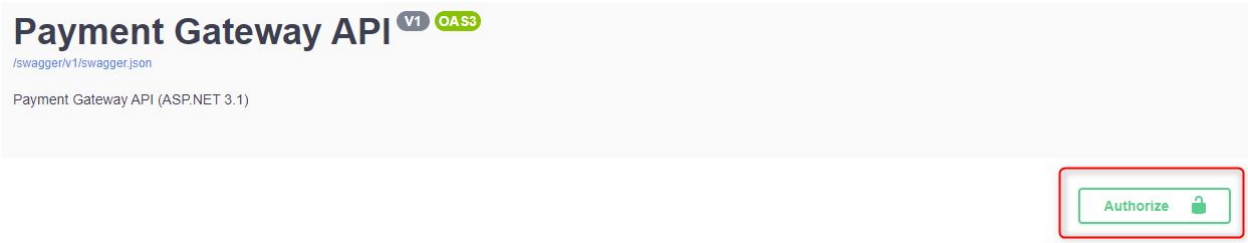

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJQZXRlciI6IlB1bXB5IiwiZXhwIjoxNjk4NTc3NTI0LCJpc3MiOiJodHRwczovL2xvY2FsaG9zdDo2MDAxIiwiYXVkIjoiaHR0cHM6Ly9sb2NhbGhvc3Q6NjAwMSJ9.8_qyzWbfrw8CfAymD7gaZJk2bw83zNc_PxkC_7-8qUE
```


Swagger UI

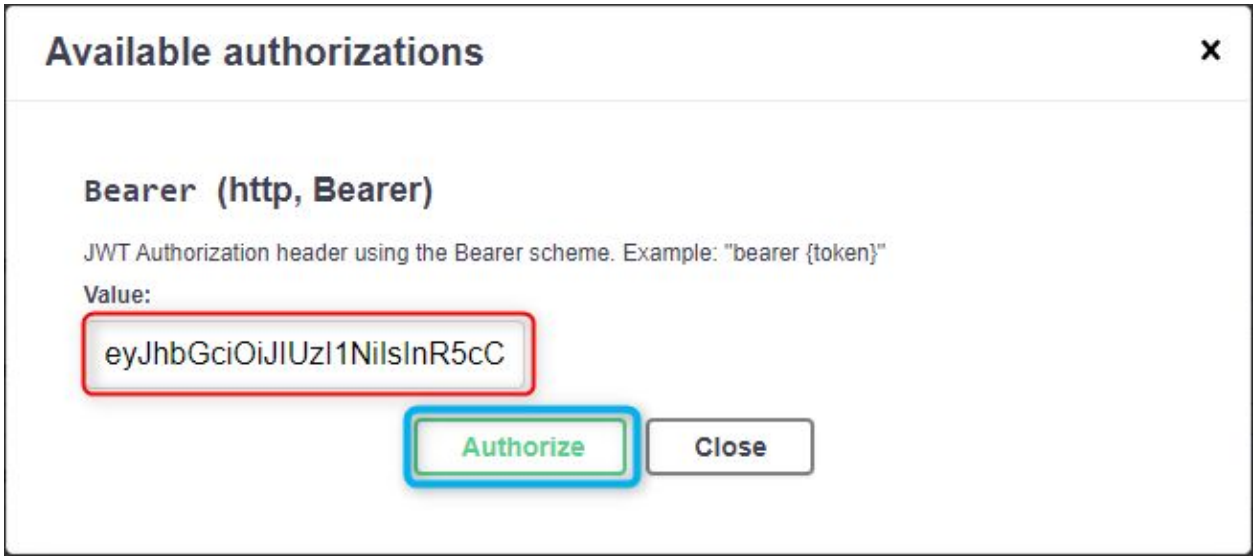
1. **Navigate to:**

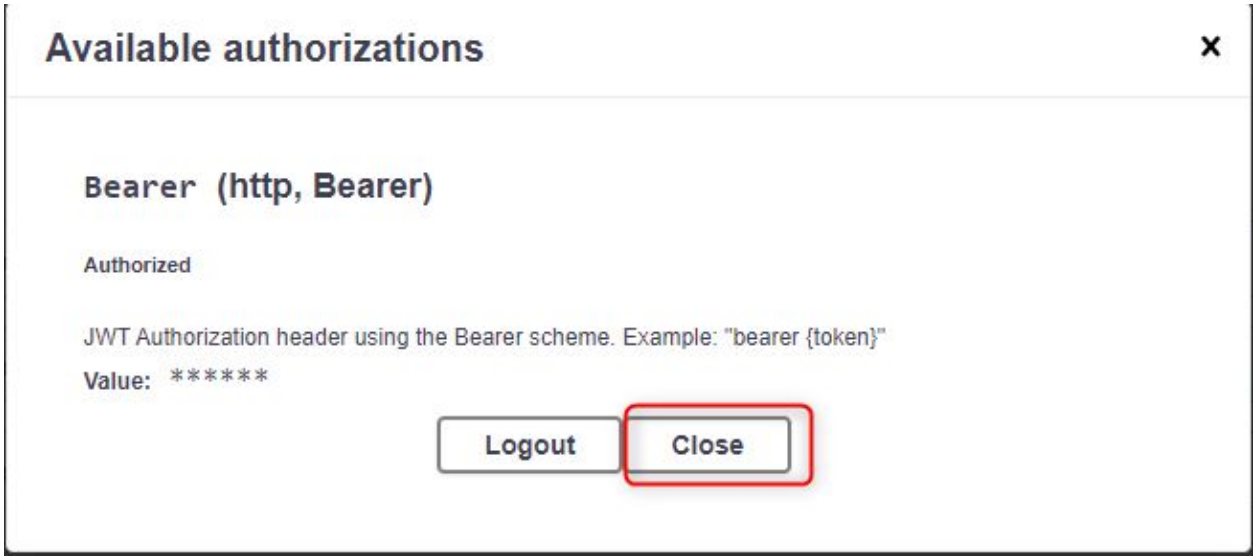
```
https://localhost:6001/Swagger/index.html
```

2. **Click Authorize**

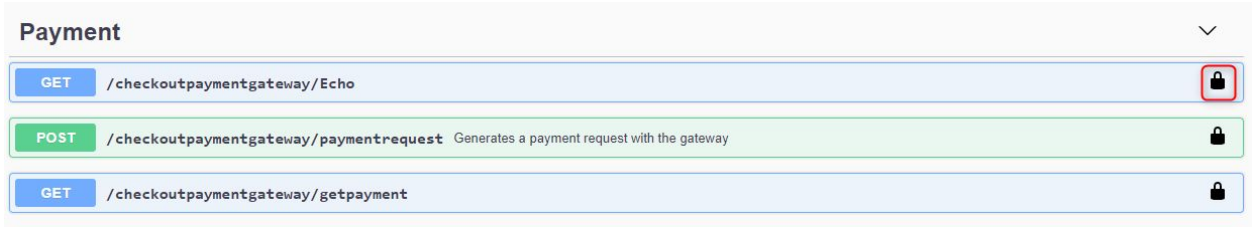


3. **Add the bearer token in to the box and click Authorize then close the popup**



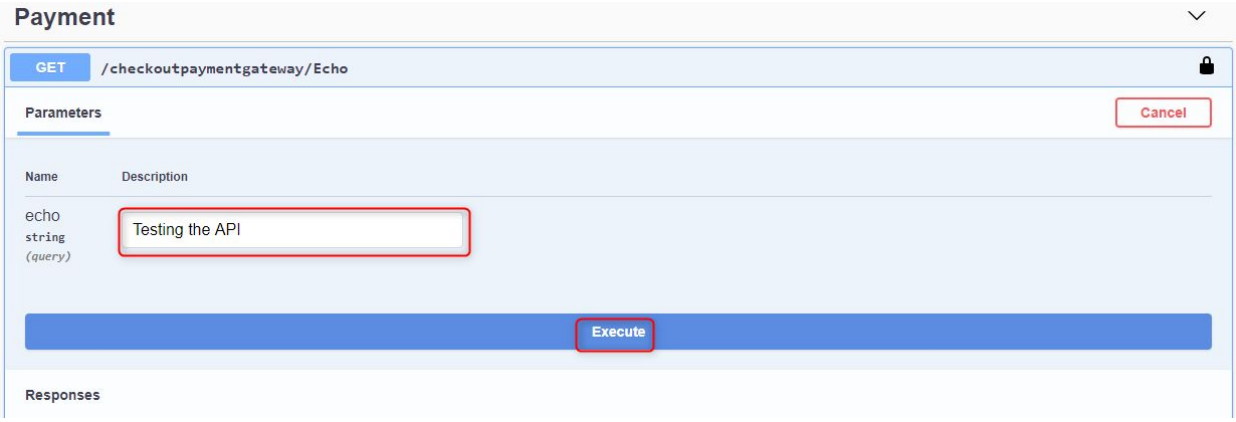


NOTE: the padlocks on the API endpoints are now closed

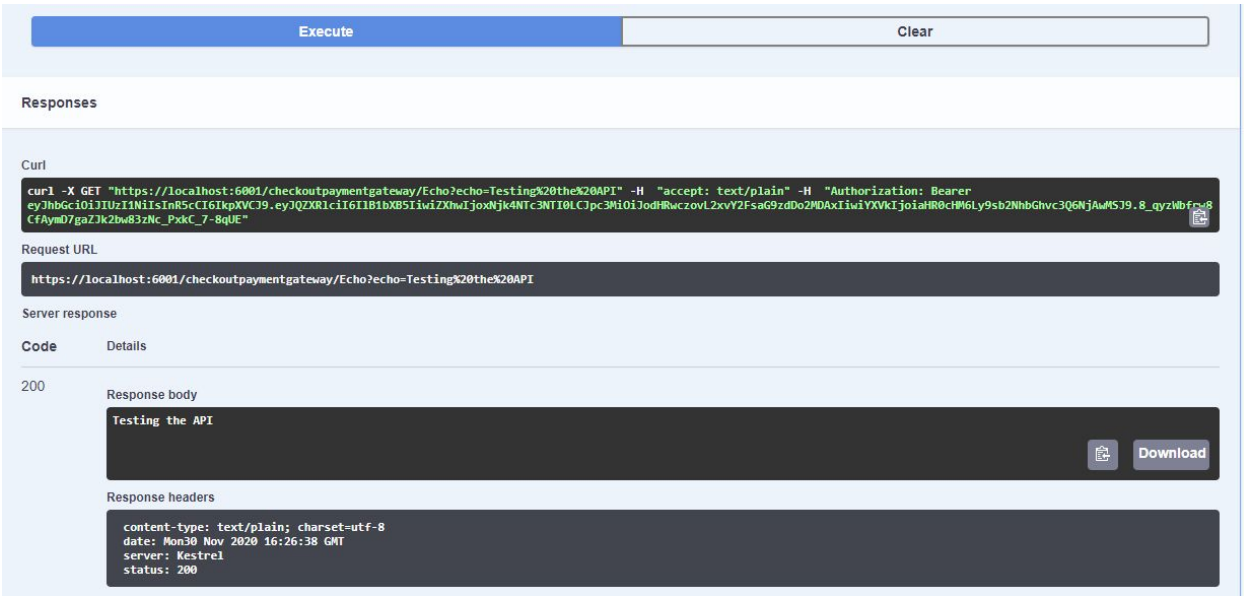


4. **Click the API Endpoint you wish to test**
5. **Click Try it out**

6. **Add a valid request (in this case any string value)**
7. **Click Execute**



8. **Check the response**

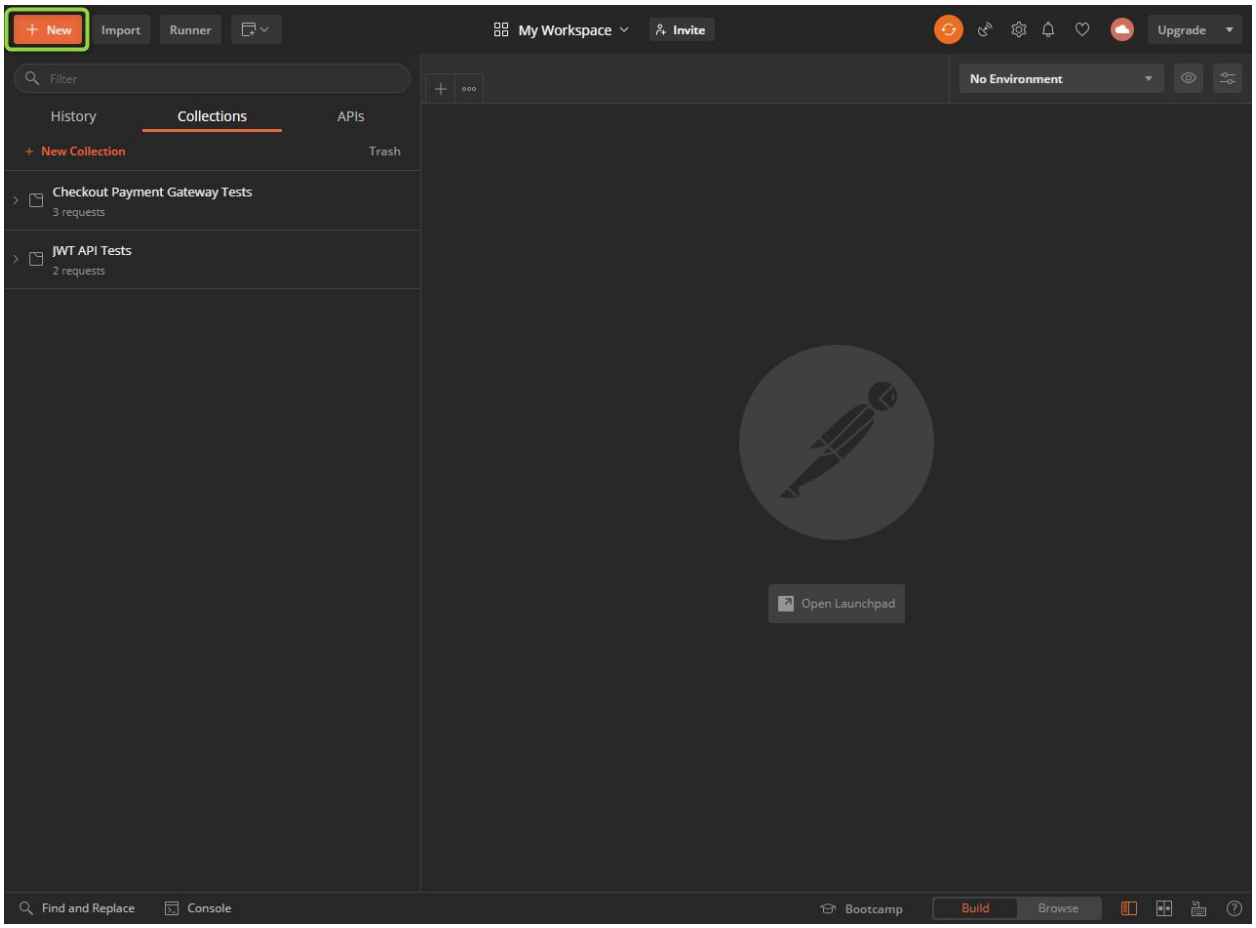


Postman

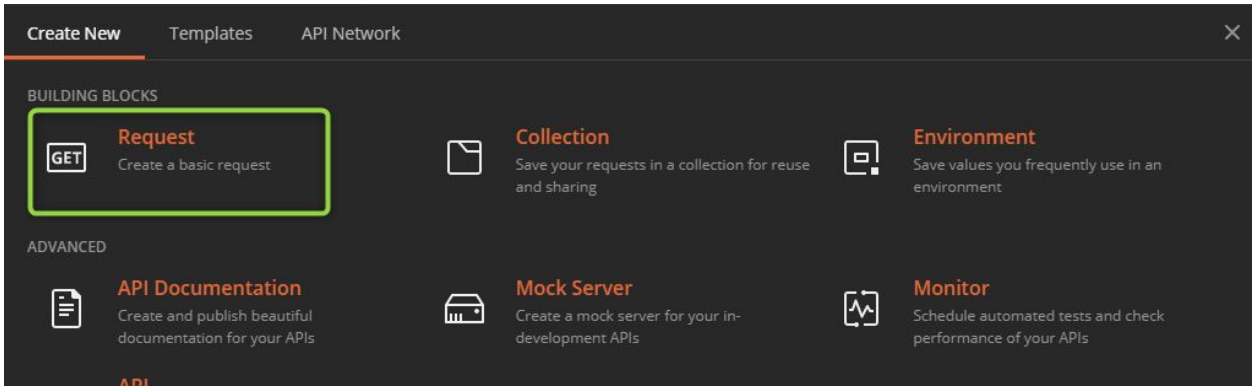
Note 1: There is a test project called PaymentGatewayAPIClientTests that can be run to get the status of the Server Endpoint

NOTE 2: The console outputs all failed payments to a JSON file to the MyDocuments folder. To change the location go to the PaymentsSetup Class and edit the Line 79.

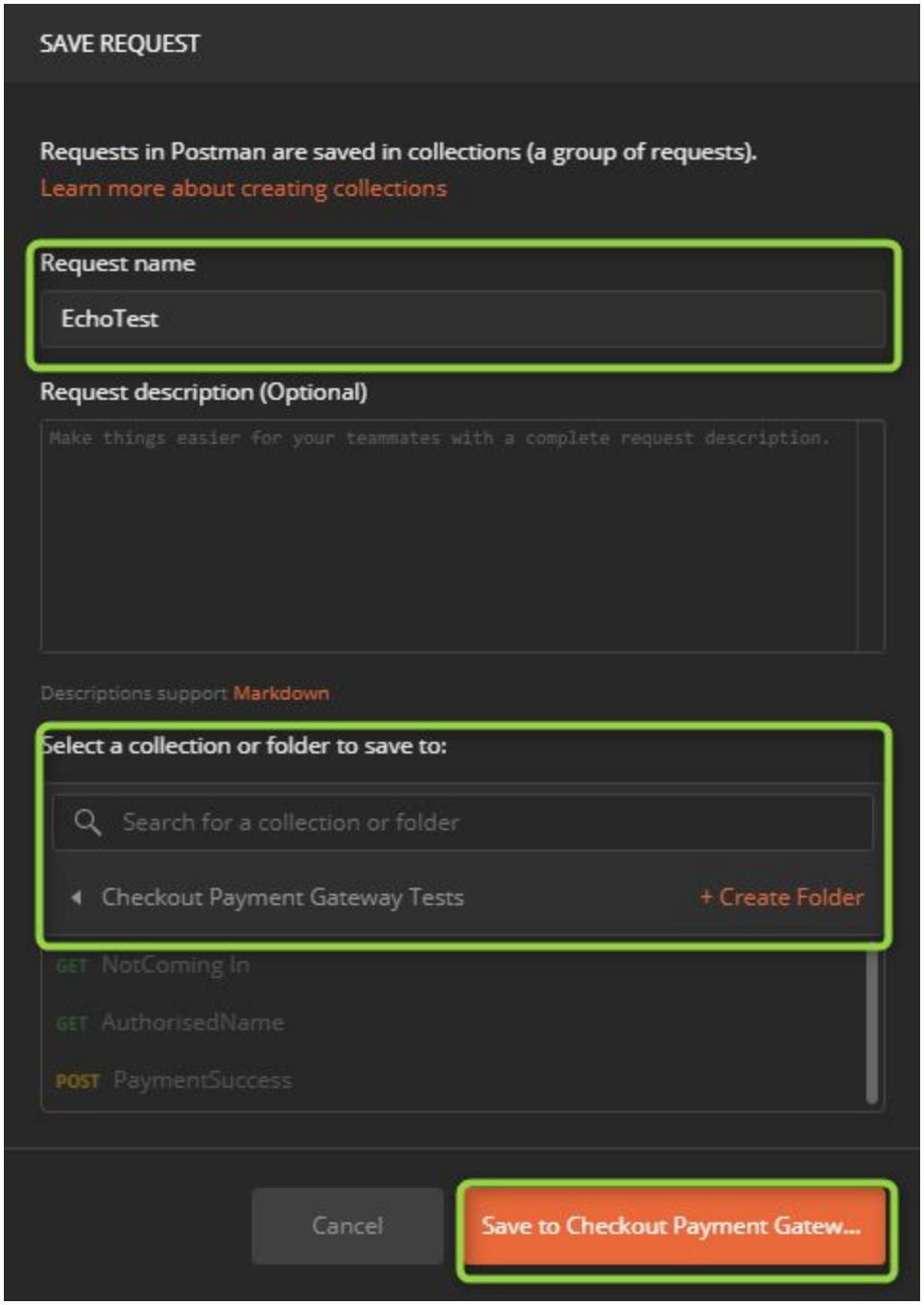
1. **Click New in the top left**



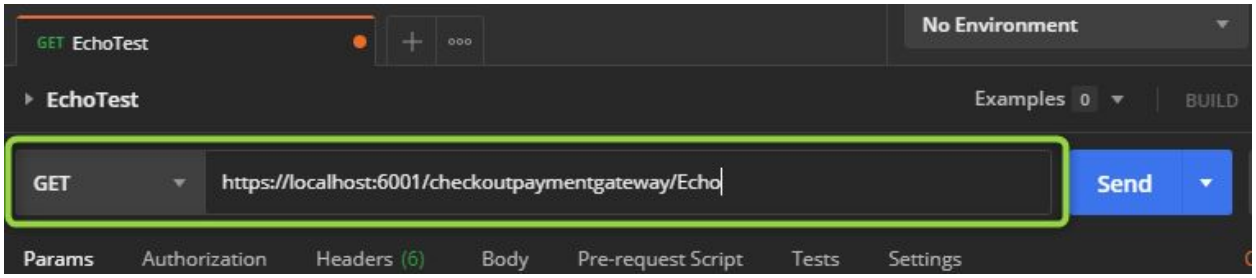
2. **Click Request**



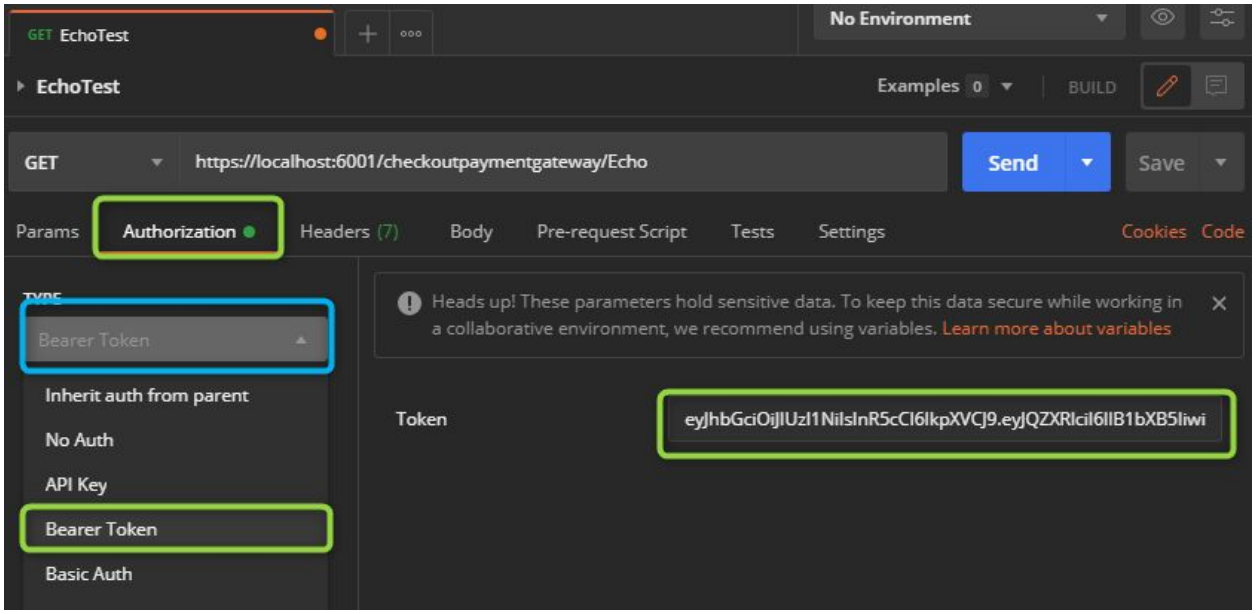
3. **Create a useful name**
4. **Add it to a collection**
5. **Save to collection**



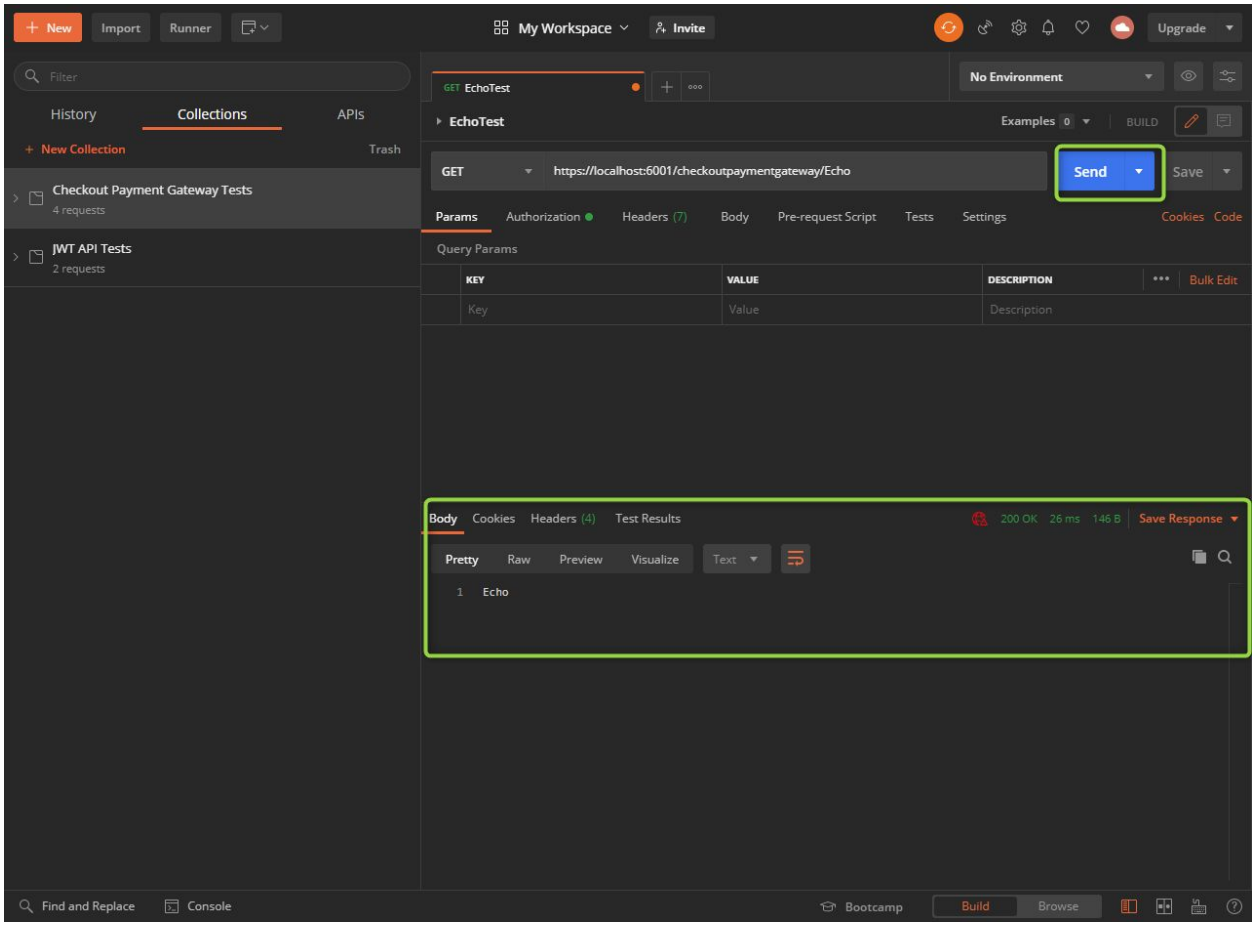
6. **Enter the endpoint URL**



7. **Click the Authorization tab**
8. **Select Bearer Token from the drop down**
9. **Enter the Bearer Token**



10. **Click Send to receive the string Echo**



Edit the URL to:

```
https://localhost:6001/checkoutpaymentgateway/Echo?echo=TestingTheApi
```

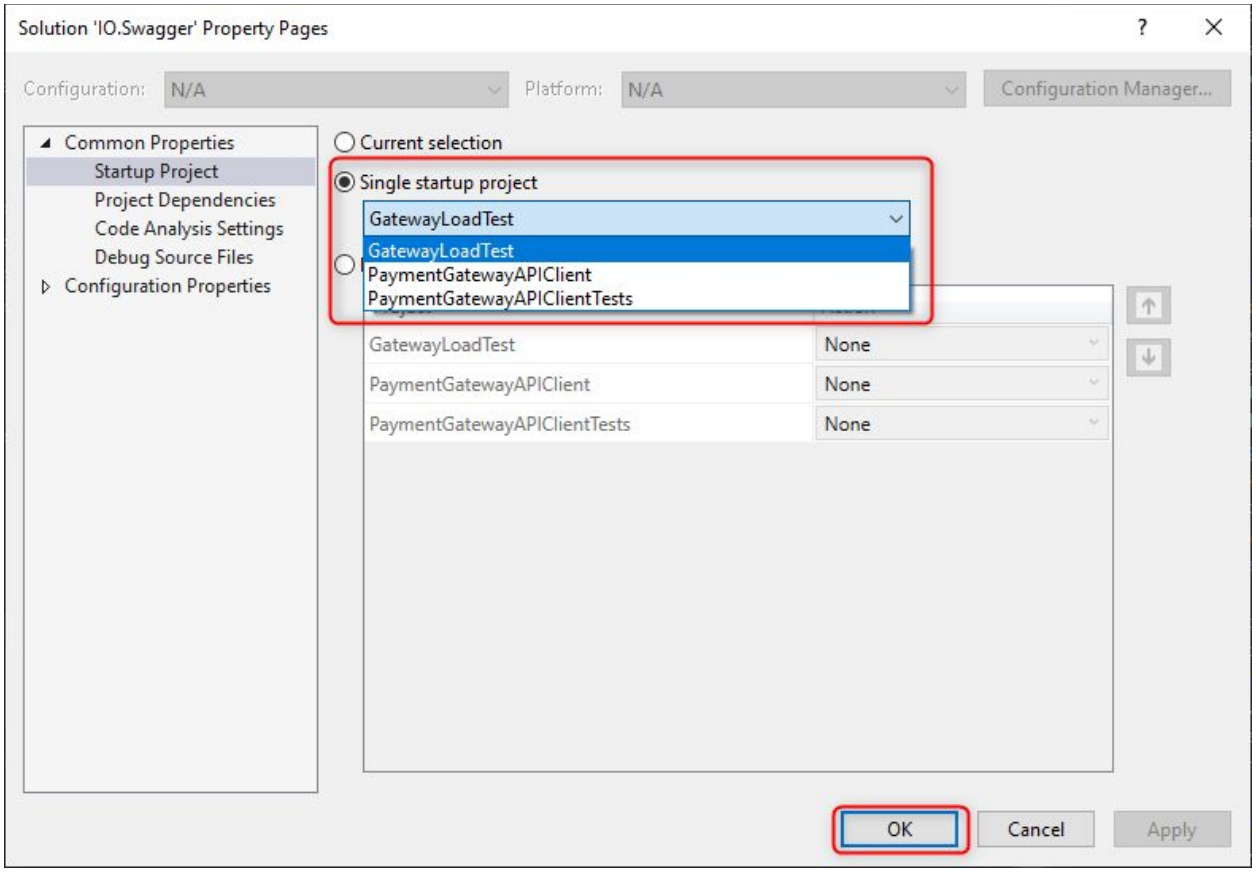
Pass your own string value to the echo endpoint

Console App in the Client Api

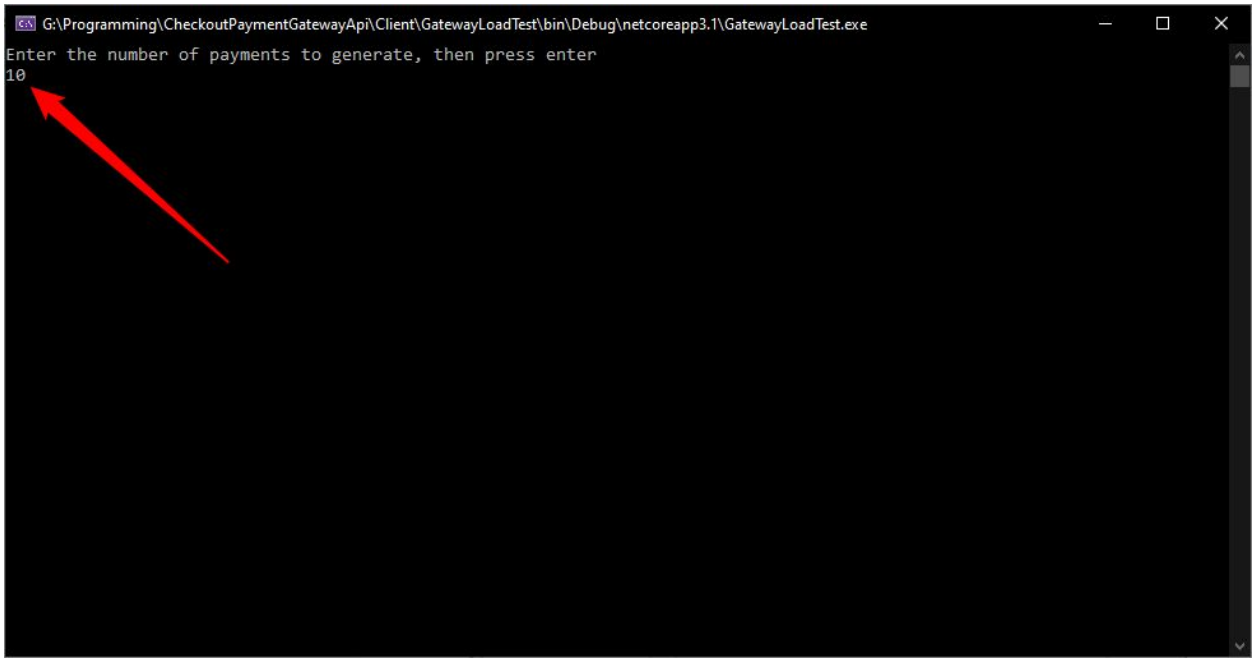
1. **Navigate to the Client folder**

```
{Path}\CheckoutPaymentGatewayApi\Server\CheckoutPaymentGateway
```

2. **Open the solution**
3. **Right click on the solution and open properties**
4. **Set GatewayLoadTest as the start-up project**



5. **Press F5 to run the project**
6. **Type the number of payments you with to send to the API**



7. **The console with show the basic breakdown of:**
    a. **Total Requests**
    b. **Total passing requests**
    c. **Total failing requests**
    d. **A file path to a JSON output file of all the failed requests**