How to use ORP in NewsREEL'17

In the 2017 edition of CLEF NewsREEL, we continue to offer participants the unique opportunity to connect their recommendation engine to an active news recommender system. This connection is established via the Open Recommendation Platform (ORP). This document introduces the RESTful Application Programming Interface (API) which has been made available to participants.

Requirements

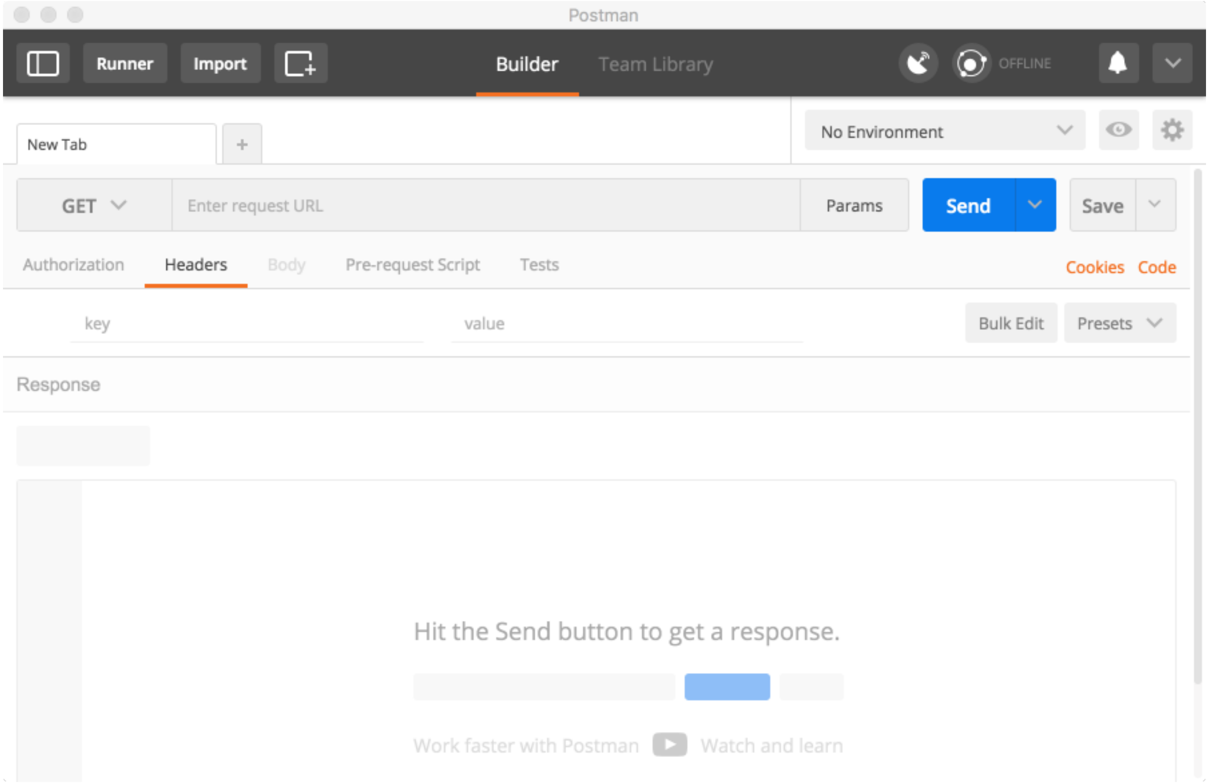
We assume the participant to meet following conditions:

- you should have received an email with user name and password from plista
- have deployed a recommendation engine which can be access via HTTP (URL and Port)

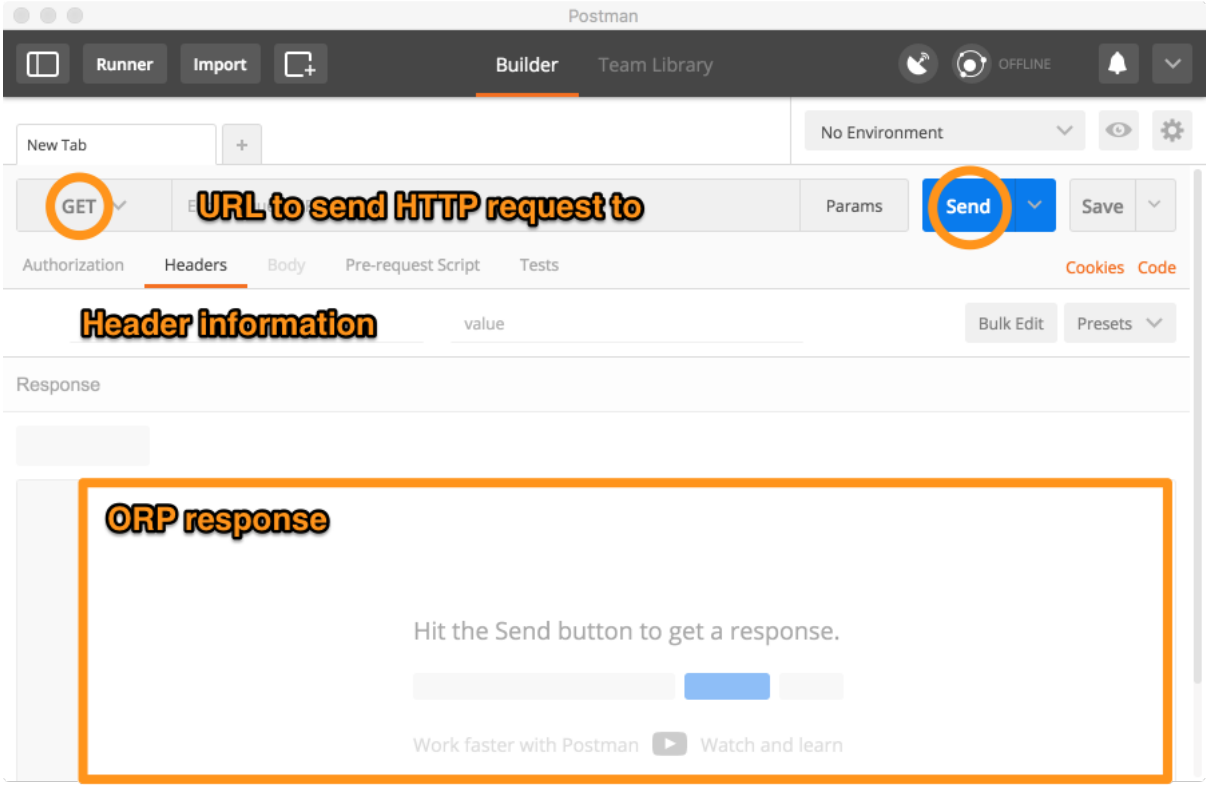
There are two options for participants to use the API. They can either issue cURL statements from the command line or use a REST Client. We use the REST Client "Postman" to illustrate how to get started.

Getting started

The following screenshot shows the start screen of Postman. Note that it is also possible to use a different REST Client, which may have a different interface. Note further, that you can use cURL to issue the calls to ORP's API from the command line as well.



For your convenience, we have explicitly marked the essential parts of the interface in another screenshot.



On the top left side there is a book that lets us select which type of request we want to issue. There are a variety of options available. We need only three of them: GET, PUT, and POST. Next, concerning the type selection, we see the field to enter the URL to which the HTTP request will be send. This will include ORP's URL (https://orp.plista.com/api/) as well as a specific addition depending on the functionality we want to use. Below we find a set of tabs, of which "Headers" is currently selected. There, we can include parameters to the call. This will mainly be used for authenticating participants. In the top right corner, we see a button, which when clicked initiates the HTTP call. ORP's response will be displayed in the larger area at the bottom.

Process Overview

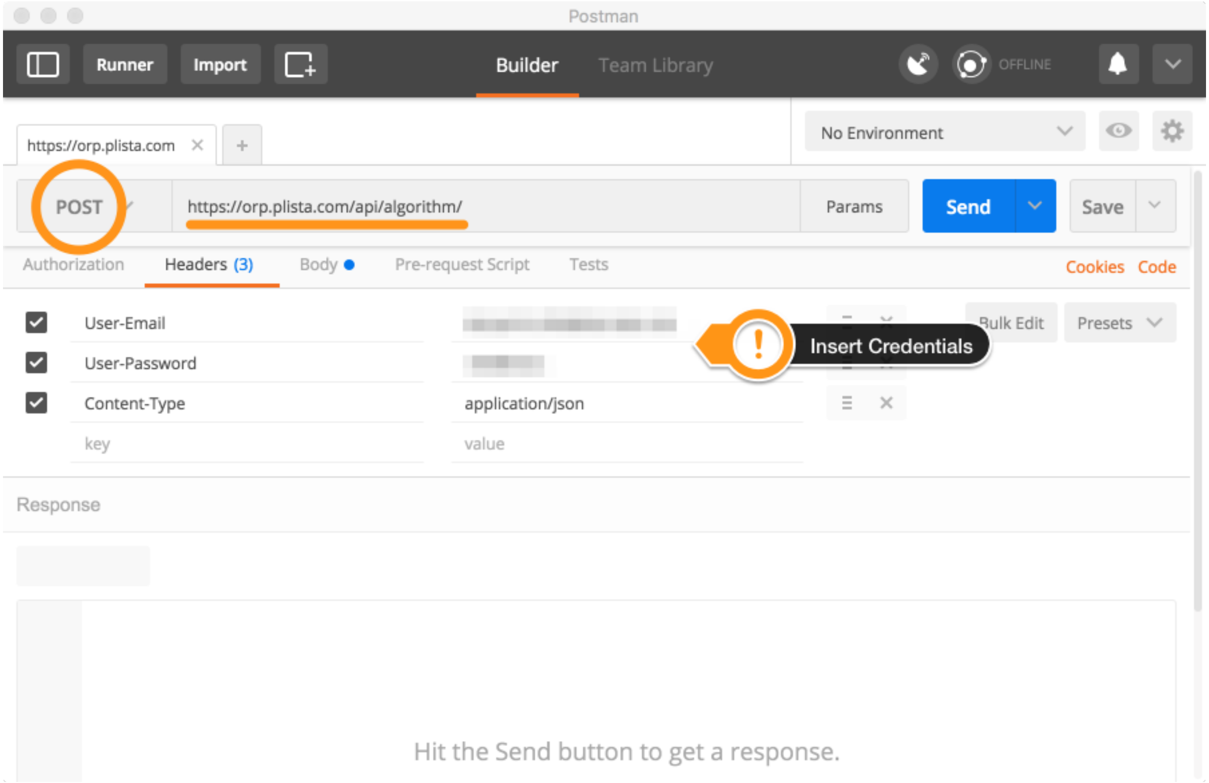
There are two steps that we have to complete to initiate the evaluation. First, we have to create an algorithm. Second, we have to activate the message stream. Having started the evaluation, we want to monitor the performance of our algorithm. At times, we may want to pause the evaluation to adjust our algorithms.

Creating an Algorithm

Creating a new algorithm involves specifying certain information. First, we select POST as type. Second, we specify the correct URL to call: https://orp.plista.com/api/algorithm/ Third, we have to attach the credentials obtained from plista to the header. We specify "User-Email" as the email address at which the credentials were originally received. Further, we specify "User-Password" with the password provided by plista. Finally, we have to provide the following information to configure the algorithm:

Parameter	Meaning
name	Reference to algorithm
itemsPath	URL to which ORP will send item updates
errorsPath	URL to which ORP will send error messages
requestsPath	URL to which ORP will send recommendation requests
notificationsPath	URL to which ORP will send notifications
status	current state of the recommender ("online" or "offline")

There parameters are specified in the body of the message. We use JSON format. The following two screenshots illustrate the settings in Postman.

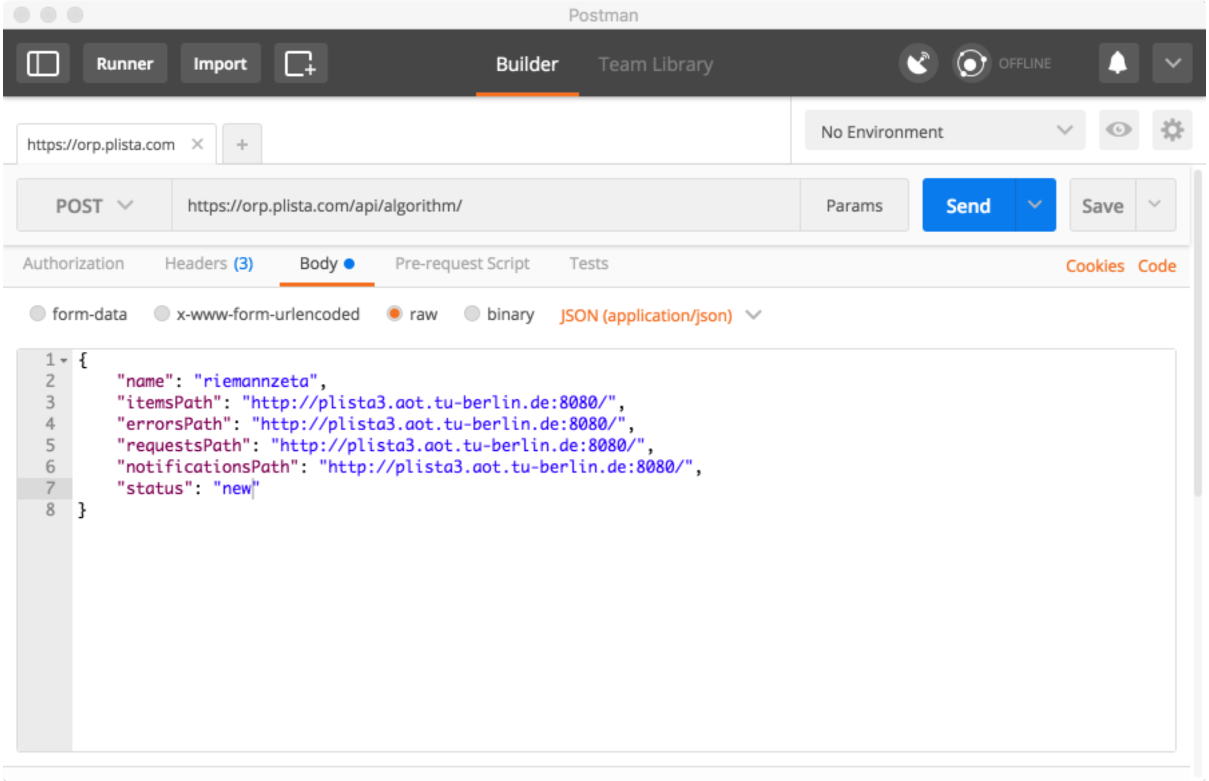


https://orp.plista.com

POST https://orp.plista.com/api/algorithm/

☑	User-Email	[redacted]
☑	User-Password	[redacted]
☑	Content-Type	application/json
	key	value

Hit the Send button to get a response.



https://orp.plista.com

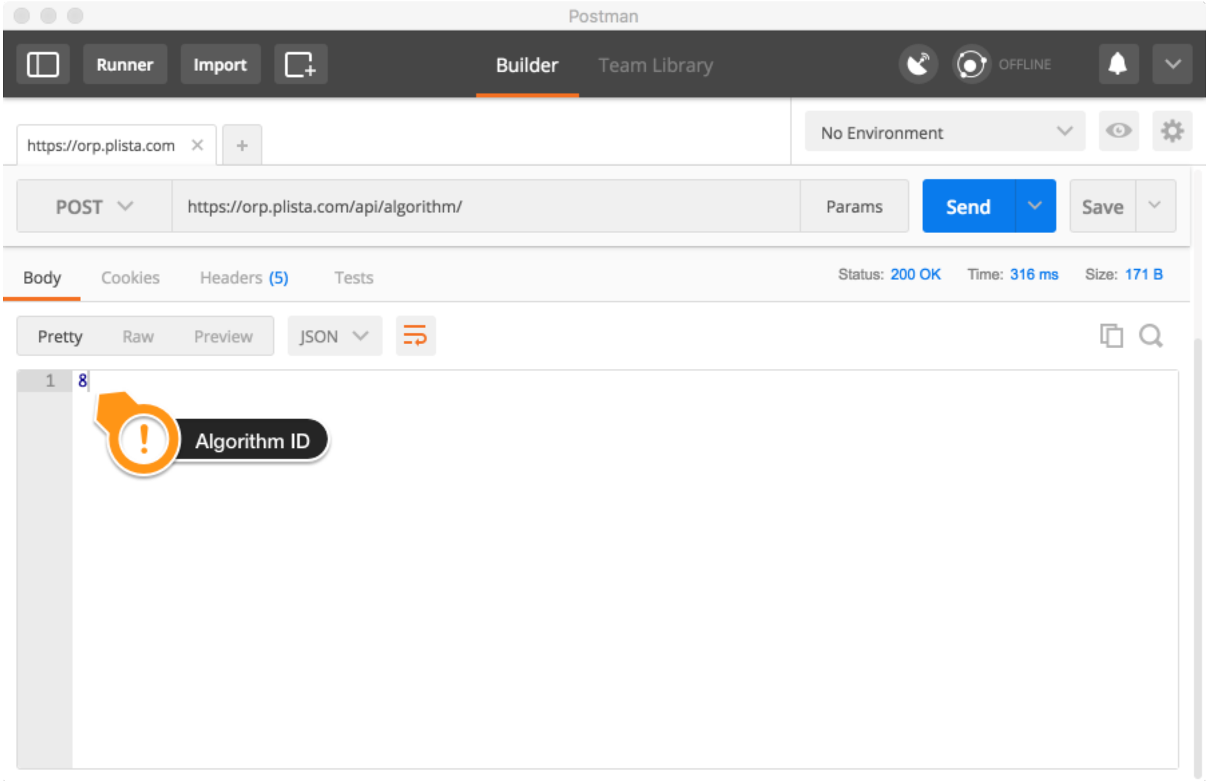
POST https://orp.plista.com/api/algorithm/

○ form-data ○ x-www-form-urlencoded ● raw ○ binary JSON (application/json)

```
{
    "name": "riemannzeta",
    "itemsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "errorsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "requestsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "notificationsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "status": "new"
}
```

In this example, we have used the same URL and Port for all types of messages emitted by ORP. Participants can also specify different URLs and/or Ports to route message types to different destinations. We have called the new algorithm “riemannzeta” and set the status to “new”.

It is crucial to record the ID returned by ORP. We need it in the future to monitor the performance as well as adjust the settings. As seen in the following screenshot, we have received ID=8 for our algorithm.



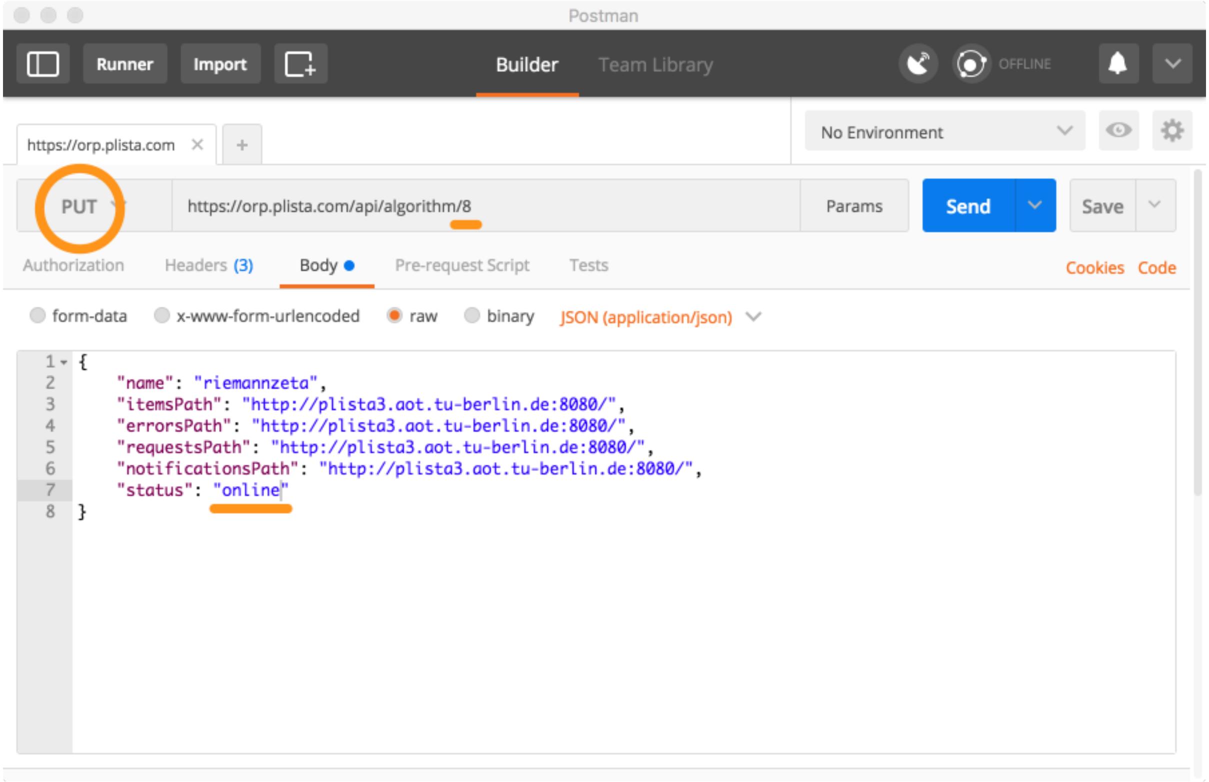
POST https://orp.plista.com/api/algorithm/

Status: 200 OK Time: 316 ms Size: 171 B

```
8
```

Activate Message Stream

Having created an algorithm, we can now move forward by activating the communication with ORP. Note that the recommendation engine has to be running and accessible. The activation procedure requires adjusting the previous call. Instead of a POST we use a PUT to update the configuration. In addition, we have to specify which recommender we want to change. In our case, we want to switch on the algorithm with ID=8. To this end, we append the previously used URL with the algorithm ID to obtain https://orp.plista.com/api/algorithm/8. Finally, we copy the previous configuration to the body and replace the “status”: “new” with “status”: “online”. ORP will start sending messages shortly after we have clicked the Send button. Note that we keep the authentication information in the header. The activation is depicted in the following screenshot:



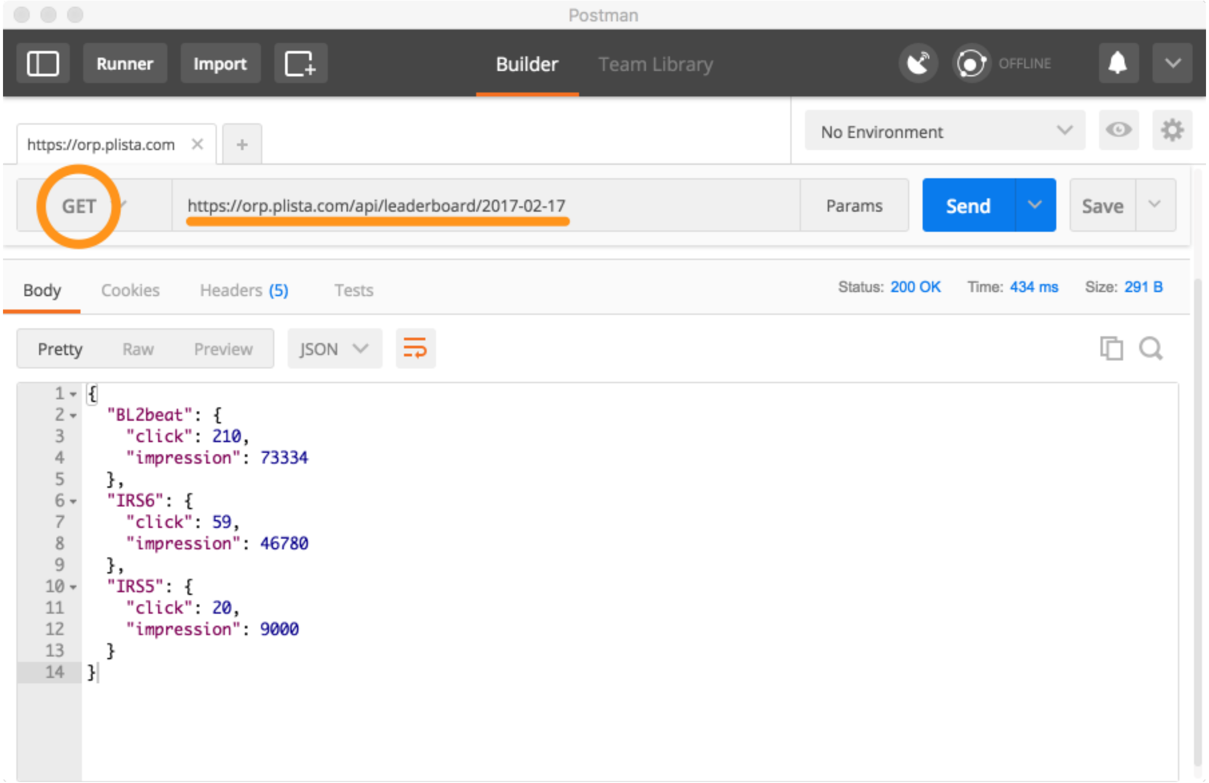
PUT https://orp.plista.com/api/algorithm/8

```
{
    "name": "riemannzeta",
    "itemsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "errorsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "requestsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "notificationPath": "http://plista3.aot.tu-berlin.de:8080/",
    "status": "online"
}
```

If anything has gone wrong we will receive an error message instead of the expected {“result”: true}.

Monitoring Performance

Participants can log all messages they receive. In general, this enables all participants to monitor their performance individually. Additionally, we would like to compare the performance of all participants. For this reason, ORP provides a functionality that makes it possible to look at the current leaderboard. The leaderboard can be accessed with a GET call to the URL https://orp.plista.com/api/leaderboard/YYYY-MM-DD. Participants can replace the date template with the start date for the evaluation. For instance, looking at the performance of all participants in the time span from 21 February 2017 to today (3 March 2017), we would call the URL https://orp.plista.com/api/leaderboard/2017-02-17. The following screenshot illustrates the exact query:



https://orp.plista.com

GET https://orp.plista.com/api/leaderboard/2017-02-17

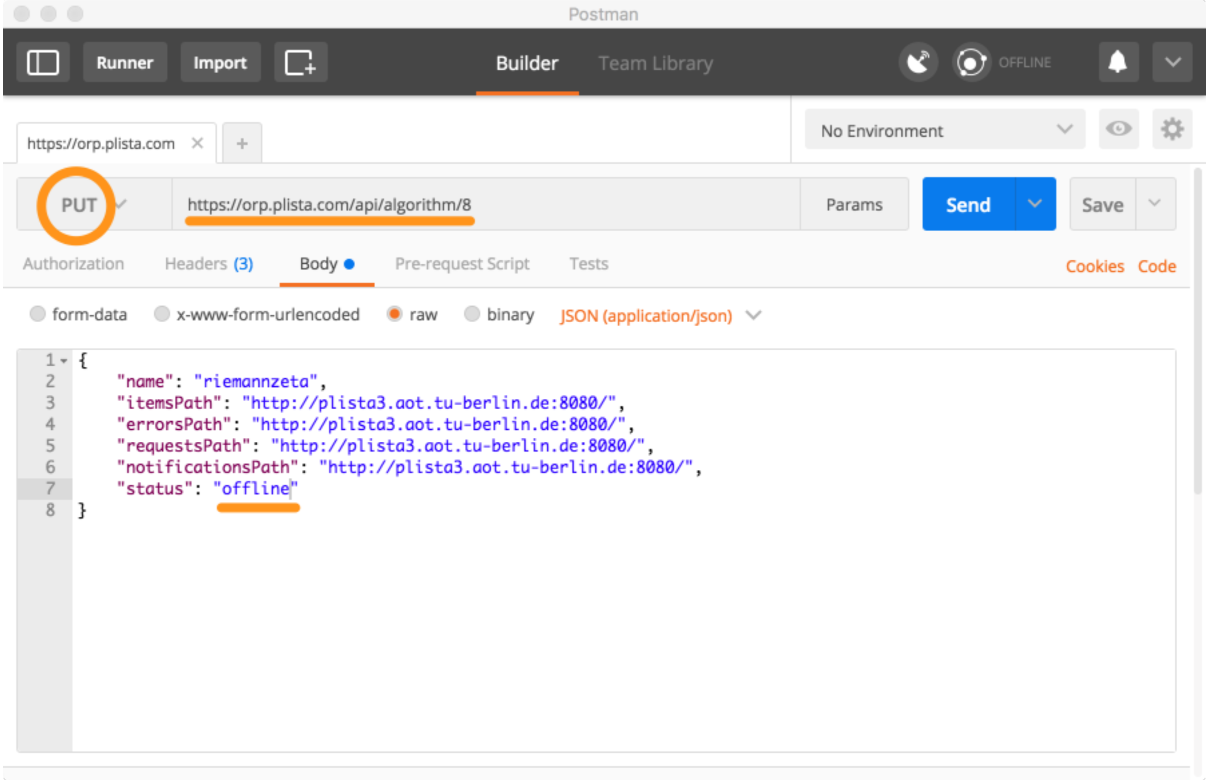
Status: 200 OK Time: 434 ms Size: 291 B

```
{
  "BL2beat": {
    "click": 210,
    "impression": 73334
  },
  "IRS6": {
    "click": 59,
    "impression": 46780
  },
  "IRS5": {
    "click": 20,
    "impression": 9000
  }
}
```

There have been three algorithms active in this time span. We receive the number of requests/impressions as well as the number of clicks for each individual algorithm. With this information we can compute the CTR and compare relative performance.

Adjusting Configuration

Whenever we want to change the algorithm, we should pause the message stream in order not to confuse the evaluation. We accomplish this in. a similar fashion as the activation. Again, we call with PUT to the URL https://orp.plista.com/api/algorithm/ID (in our case we would replace ID with 8). This time we set the status to “offline” instead of “online”. After a while, ORP will stop sending messages. Then we can adjust the recommender and activate it again as described previously. The deactivation procedure is shown in the following screenshot:



PUT https://orp.plista.com/api/algorithm/8

○ form-data ○ x-www-form-urlencoded ● raw ○ binary JSON (application/json)

```
{
    "name": "riemannzeta",
    "itemsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "errorsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "requestsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "notificationsPath": "http://plista3.aot.tu-berlin.de:8080/",
    "status": "offline"
}
```

Additional Information

You may find additional information in ORP's documentation (https://orp.plista.com/documentation/download —> specification of the information sent to your recommender), ORP's API documentation (https://orp.plista.com/api/documentation) or on the NewsREEL website (http://clef-newsreel.org).
You can access the source code of the baseline implementation at https://github.com/andreas-dai/NewsREEL-Template
If you have further questions that are not covered here, please, contact the organisers.

Frank Hopfgartner (frank.hopfgartner@glasgow.ac.uk)
Benjamin Kille (benjamin.kille@tu-berlin.de)
Andreas Lommatzsch (andreas.lommatzsch@dai-labor.de)
Martha Larson (m.a.larson@tudelft.nl)

Appendix

For participants preferring to use cURL, we include the equivalent API calls.

1. Creating a new algorithm:

```
curl \
        -X POST \
        -H “User-Email: name@domain.org" \
        -H “User-Password: 123456” \
        -H “Content-Type: application/json” \
        -H “Accept: text/html” \
        -d ‘{
                “name”: “riemannzeta”,
                “itemsPath”: “http://plista3.aot.tu-berlin.de:8080”,
                “errorsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “requestsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “notificationsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “status”: “new”
        }’ \
        https://orp.plista.com/api/algorithm
```

2. Activate message stream:

```
curl \
        -X PUT \
        -H “User-Email: name@domain.org" \
        -H “User-Password: 123456” \
        -H “Content-Type: application/json” \
        -H “Accept: text/html” \
        -d ‘{
                “name”: “riemannzeta”,
                “itemsPath”: “http://plista3.aot.tu-berlin.de:8080”,
                “errorsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “requestsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “notificationsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “status”: “online”
        }’ \
        https://orp.plista.com/api/algorithm/8
```

3. Monitoring performance:

```
curl \
        -X GET \
        -H “User-Email: name@domain.org" \
        -H “User-Password: 123456” \
https://orp.plista.com/api/leaderboard/2017-02-17
```

4. Adjusting configuration:

```
curl \
        -X PUT \
        -H “User-Email: name@domain.org" \
        -H “User-Password: 123456” \
        -H “Content-Type: application/json” \
        -H “Accept: text/html” \
        -d ‘{
                “name”: “riemannzeta”,
                “itemsPath”: “http://plista3.aot.tu-berlin.de:8080”,
                “errorsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “requestsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “notificationsPath”: “http://plista3.aot.tu-berlin.de:8080",
                “status”: “offline”
        }’ \
        https://orp.plista.com/api/algorithm/8
```