This document aims to guide you how to create a new task in EcoPAD and publish it on the website. We will start with how to build a docker image, and then how to connect the docker image with the EcoPAD platform.

Sample Fortran code

The following is a very simple Fortran code with the input from command line. In this example, you will learn how the code receive the arguments from command line. Copy the code to an empty file (**e.g., test.f90**).

```
program main

    character(len=10) first_command_line_argument   !declaration
    character(len=10) second_command_line_argument  !declaration

    call getarg(1, first_command_line_argument)     !get argument
    call getarg(2, second_command_line_argument)    !get argument

    print *, first_command_line_argument            !print
    print *, second_command_line_argument           !print

end program
```

Compile the file with the following command:

```
gfortran -o test.o test.f90
```

Run the executable file with the following command:

```
./test.o test1 test2
```

The output should be:

```
 test1
 test2
```

Fortran utilizes the function `getarg()` to receive the arguments from command line. In the model, the paths to some crucial files are usually passed by command line arguments. Thus, it is important for us tu know how it works.

Build a docker image

In the section, you will learn how to build a docker image together with the sample code showing above. To get started, make sure you have docker installed on your computer. Create a new file named `Dockerfile` in the same folder as test.f90.

Dockerfile

```
From ubuntu:18.04
RUN apt-get update
RUN apt-get install -y gfortran
COPY test.f90 /root/
WORKDIR /root
RUN gfortran -o test.o test.f90
```

Run the following command to create a new docker image:

```
docker build -t test:latest .
```

Once finished, you may run the docker image with:

```
docker run test ./test.o test1 test2
```

In the command, `test` is the docker image we justed created. `./test.o` is the executable file in the `WORKDIR` of the docker image. `test1` and `test2` are two command line arguments.

Create a new task in EcoPAD

This section will guide you how to create a new task in EcoPAD, which is very essential to further develop the platform.

Every function create in the file tasks.py with the decorator `@task()` can be recognized as a task in EcoPAD. You may create a new function as follows:

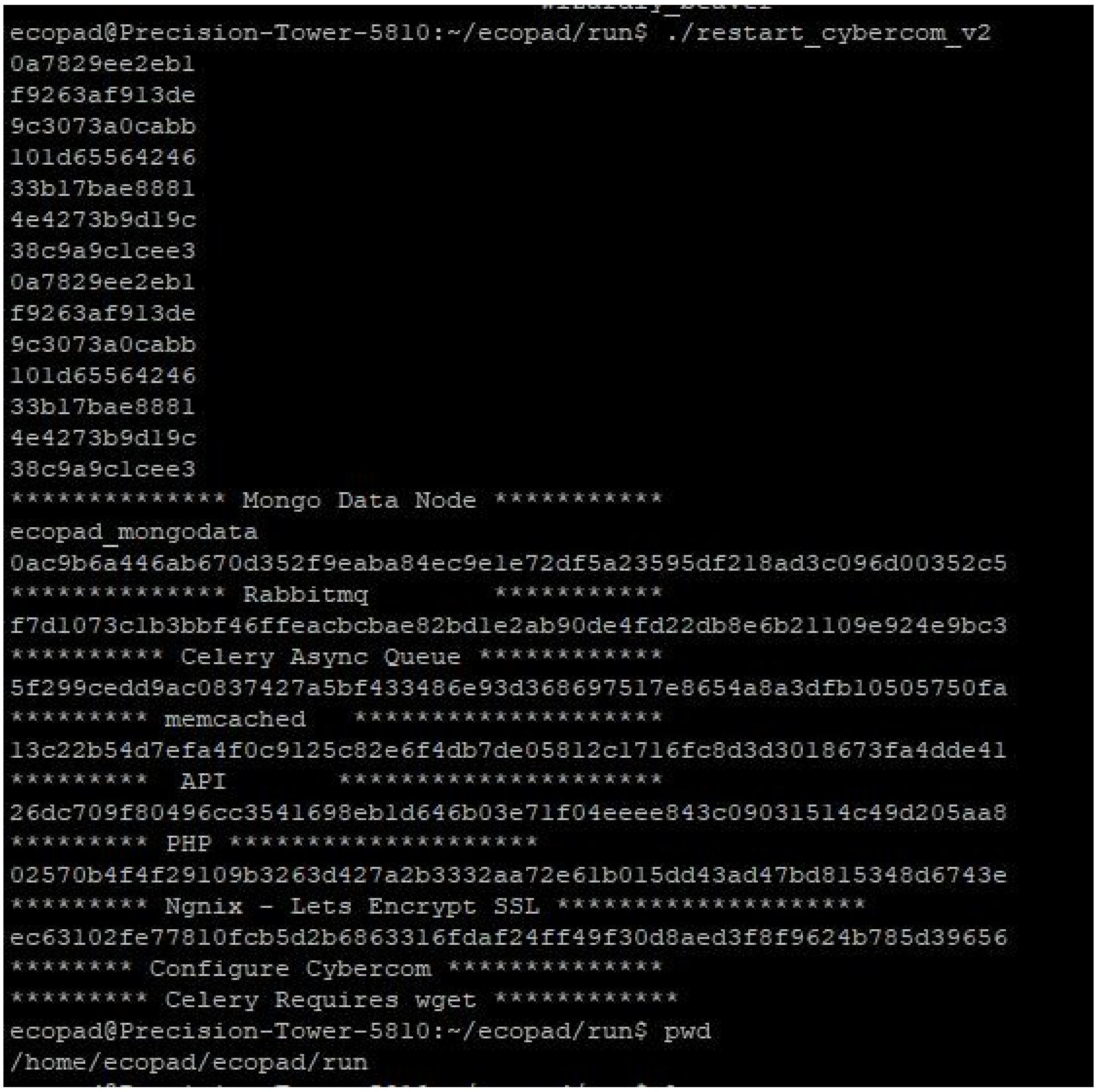
```
@task()
def test(pars):
    task_id = str(test.request.id)
    input_a = pars["test1"]
    input_b = pars["test2"]
    docker_opts = None
    docker_cmd = "./test.o {0} {1}".format(input_a, input_b)
    result = docker_task(docker_name="test", docker_opts=None,
docker_command=docker_cmd, id=task_id)
    return input_a + input_b
```

Restart the services

Run the following commands to restart the services before you can run your task:

```
cd /home/ecopad/ecopad/run/
./restart_cybercom_v2
```

The output would be similar to what is shown in the following figure:



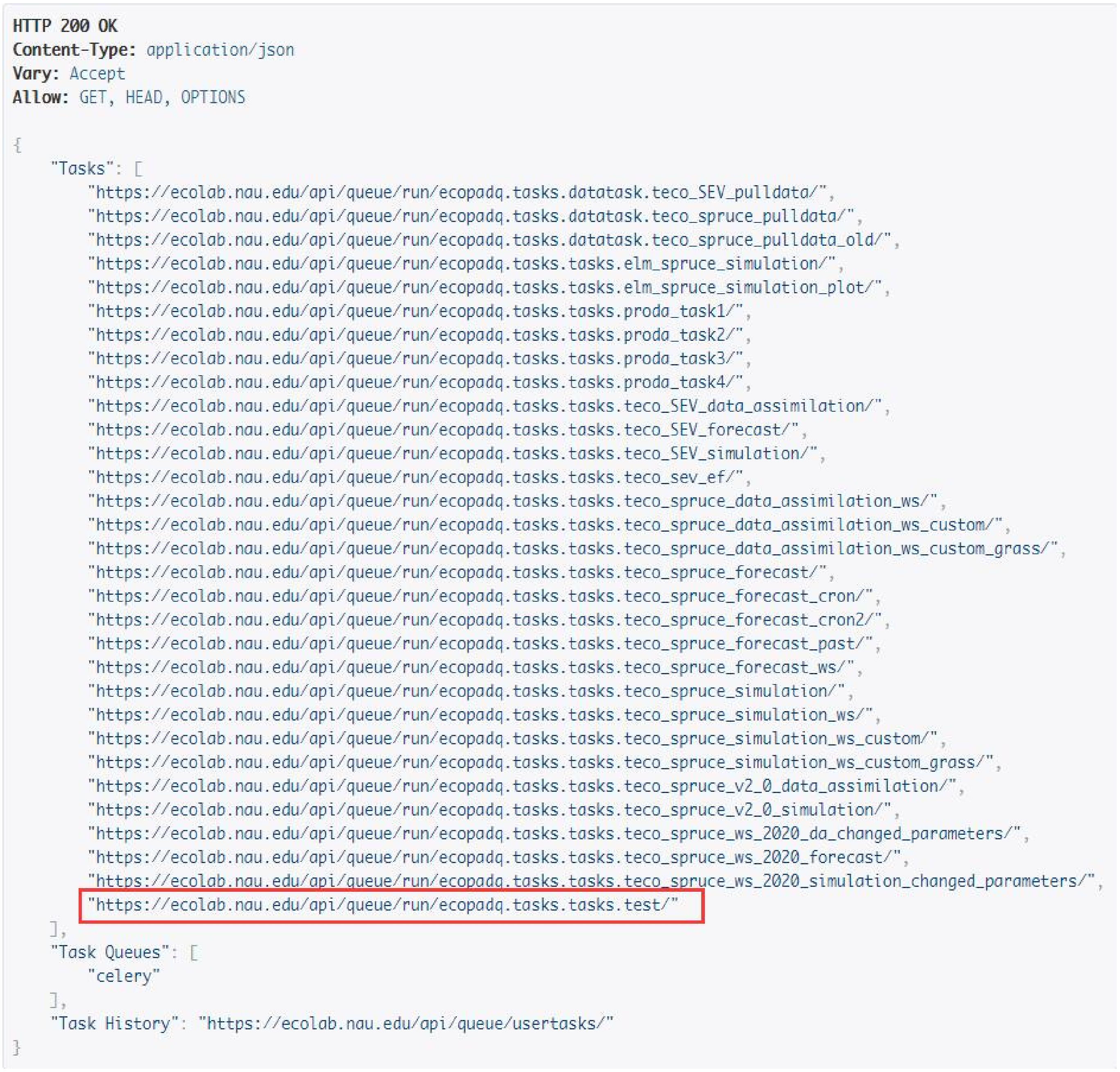
```
ecopad@Precision-Tower-5810:~/ecopad/run$ ./restart_cybercom_v2
0a7829ee2eb1
f9263af913de
9c3073a0cabb
101d65564246
33b17bae8881
4e4273b9d19c
38c9a9c1cee3
0a7829ee2eb1
f9263af913de
9c3073a0cabb
101d65564246
33b17bae8881
4e4273b9d19c
38c9a9c1cee3
************** Mongo Data Node ***********
ecopad_mongodata
0ac9b6a446ab670d352f9eaba84ec9e1e72df5a23595df218ad3c096d00352c5
************** Rabbitmq        ***********
f7d1073c1b3bbf46ffeacbcbae82bd1e2ab90de4fd22db8e6b21109e924e9bc3
********** Celery Async Queue ************
5f299cedd9ac0837427a5bf433486e93d368697517e8654a8a3dfb10505750fa
********* memcached   ********************
13c22b54d7efa4f0c9125c82e6f4db7de05812c1716fc8d3d3018673fa4dde41
*********  API       *********************
26dc709f80496cc3541698eb1d646b03e71f04eeee843c09031514c49d205aa8
********* PHP ********************
02570b4f4f29109b3263d427a2b3332aa72e61b015dd43ad47bd815348d6743e
********* Ngnix - Lets Encrypt SSL ********************
ec63102fe77810fcb5d2b6863316fdaf24ff49f30d8aed3f8f9624b785d39656
******** Configure Cybercom **************
********* Celery Requires wget ************
ecopad@Precision-Tower-5810:~/ecopad/run$ pwd
/home/ecopad/ecopad/run
```

The very long strings in the figures are the IDs of each container. If there's anything wrong in the code you just added in the file, probably the celery queue will not be able to start. You need to use the following command to see the logs and then fix the bugs in your code and then restart the services again.

```
docker logs ID
```

Run the created task

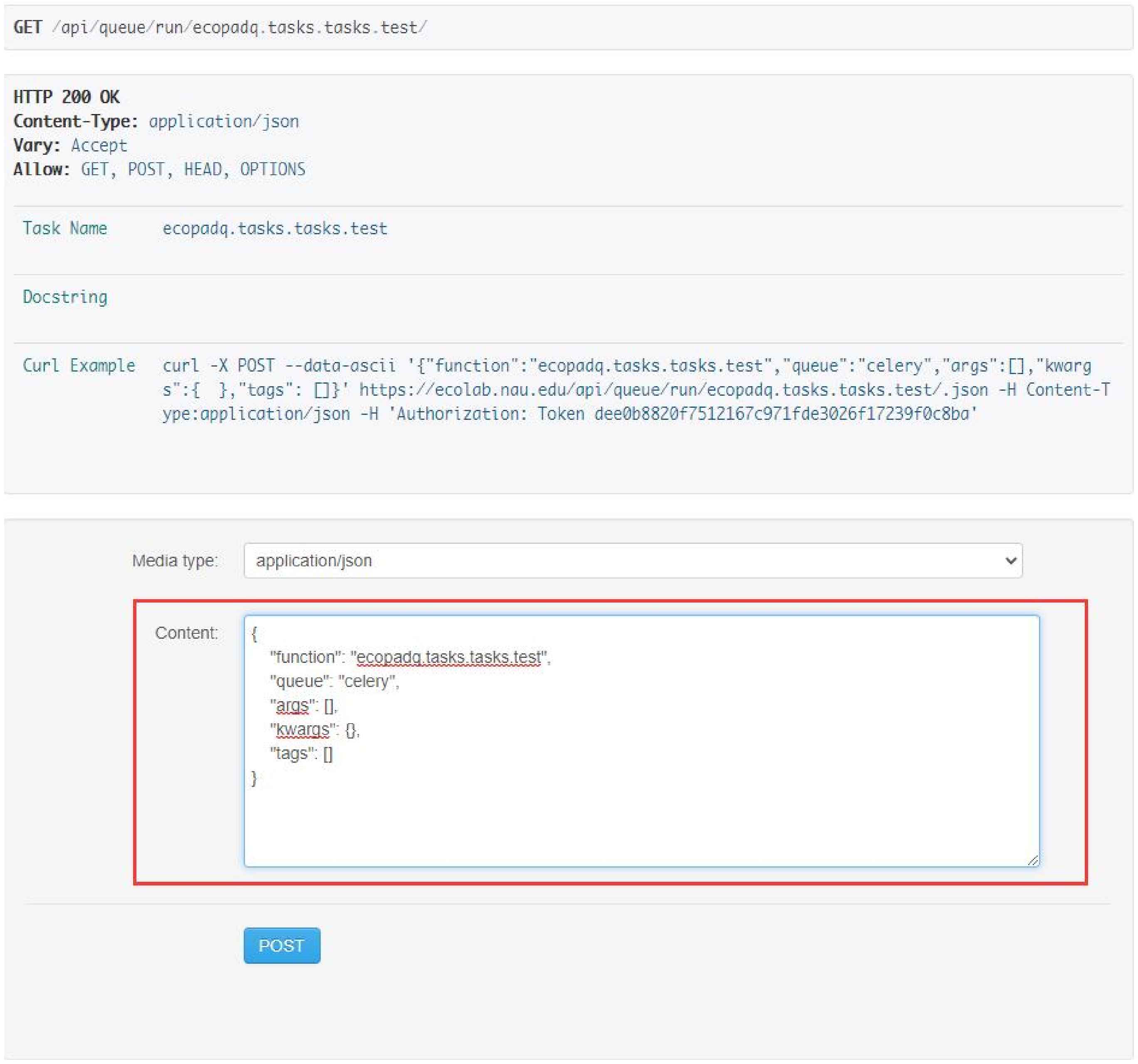
Go to this page and login with the username `ecopad` and password `ecolab02`. Once you have logged in, you will see the following page:



```
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS

{
    "Tasks": [
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.datatask.teco_SEV_pulldata/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.datatask.teco_spruce_pulldata/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.datatask.teco_spruce_pulldata_old/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.elm_spruce_simulation/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.elm_spruce_simulation_plot/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.proda_task1/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.proda_task2/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.proda_task3/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.proda_task4/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_SEV_data_assimilation/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_SEV_forecast/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_SEV_simulation/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_sev_ef/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_data_assimilation_ws/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_data_assimilation_ws_custom/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_data_assimilation_ws_custom_grass/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_forecast/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_forecast_cron/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_forecast_cron2/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_forecast_past/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_forecast_ws/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_simulation/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_simulation_ws/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_simulation_ws_custom/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_simulation_ws_custom_grass/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_v2_0_data_assimilation/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_v2_0_simulation/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_ws_2020_da_changed_parameters/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_ws_2020_forecast/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.teco_spruce_ws_2020_simulation_changed_parameters/",
        "https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.test/"
    ],
    "Task Queues": [
        "celery"
    ],
    "Task History": "https://ecolab.nau.edu/api/queue/usertasks/"
}
```

You may find the API just created in the red box. Click on that link and you will jump to another page showing below:



```
GET /api/queue/run/ecopadq.tasks.tasks.test/
```

```
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
```

Task Name	ecopadq.tasks.tasks.test
Docstring	
Curl Example	curl -X POST --data-ascii '{"function":"ecopadq.tasks.tasks.test","queue":"celery","args":[],"kwargs":{ },"tags": []}' https://ecolab.nau.edu/api/queue/run/ecopadq.tasks.tasks.test/.json -H Content-Type:application/json -H 'Authorization: Token dee0b8820f7512167c971fde3026f17239f0c8ba'

Media type: application/json

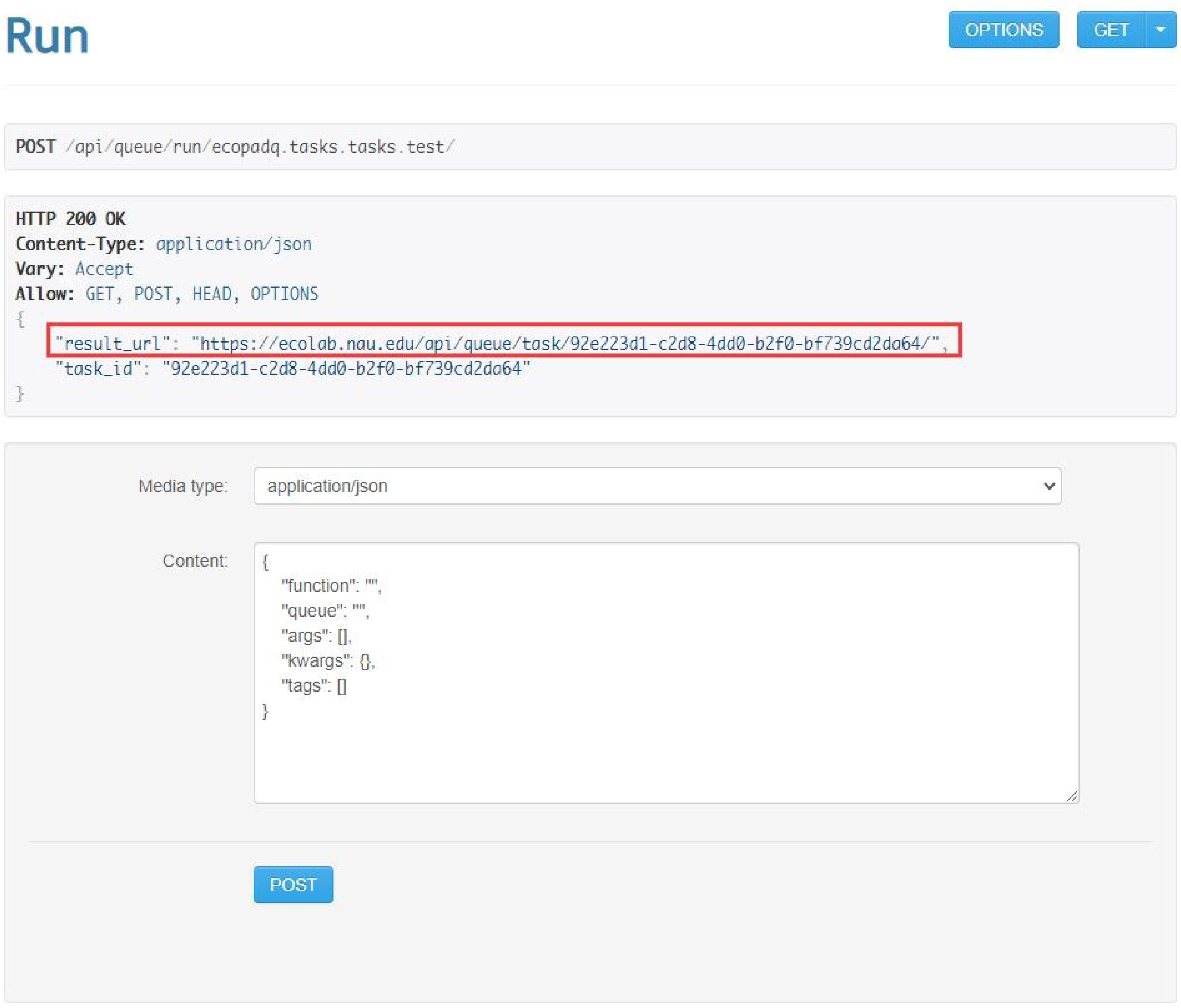
Content:

```
{
    "function": "ecopadq.tasks.tasks.test",
    "queue": "celery",
    "args": [],
    "kwargs": {},
    "tags": []
}
```

Change the content part with the following code and click POST.

```
{
    "function": "ecopadq.tasks.tasks.test",
    "queue": "celery",
    "args": [{"test1":"Hello ", "test2":"World"}],
    "kwargs": {},
    "tags": []
}
```

Once you clicked the POST button, the page will be partly refreshed. You may click the link showing in the following figure:



```
POST /api/queue/run/ecopadq.tasks.tasks.test/
```

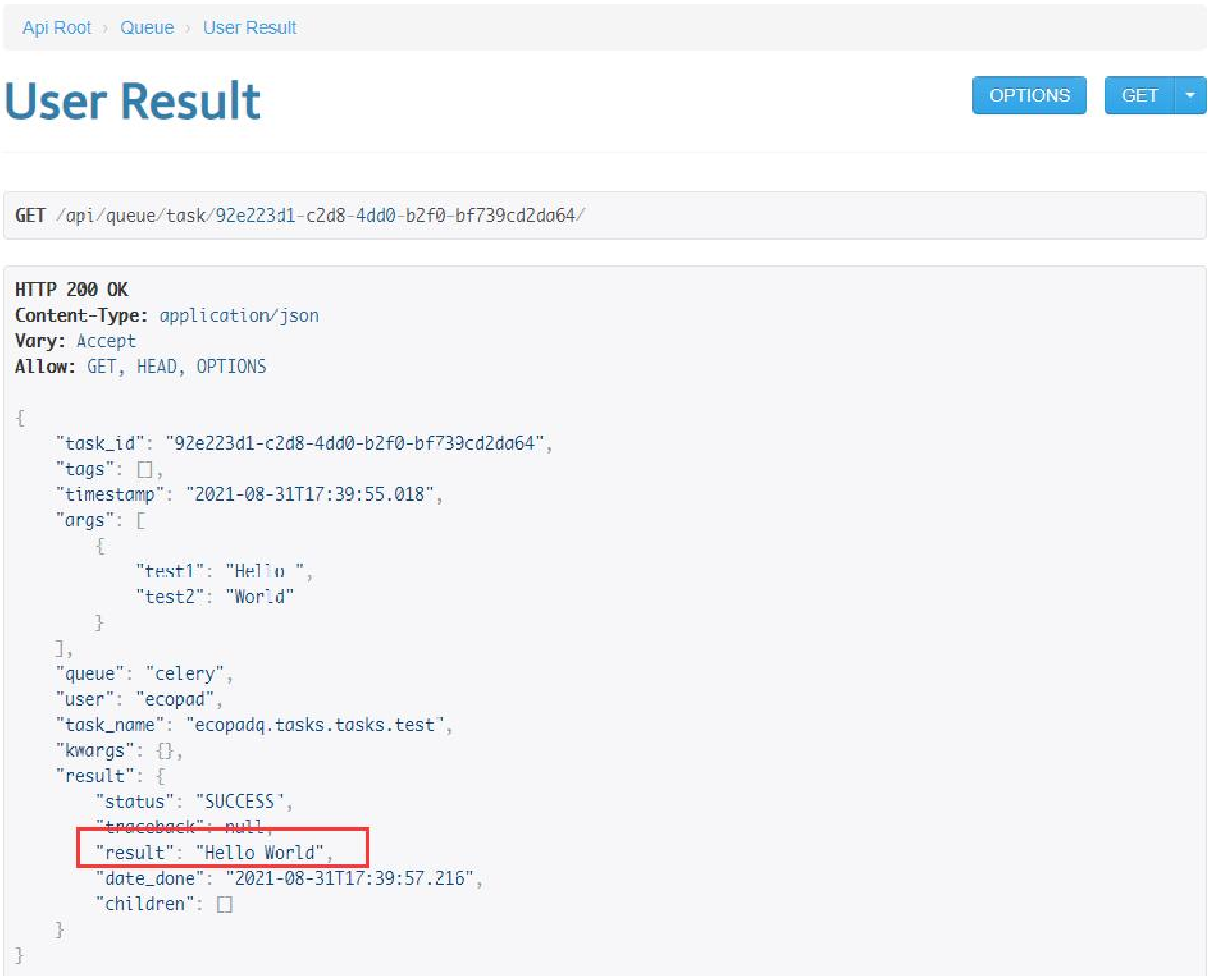
```
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, POST, HEAD, OPTIONS
{
    "result_url": "https://ecolab.nau.edu/api/queue/task/92e223d1-c2d8-4dd0-b2f0-bf739cd2da64/",
    "task_id": "92e223d1-c2d8-4dd0-b2f0-bf739cd2da64"
}
```

Media type: application/json

Content:

```
{
    "function": "",
    "queue": "",
    "args": [],
    "kwargs": {},
    "tags": []
}
```

You will jump to the result page once you clicked the button. You will find the parameters you just passed to the back-end.


Api Root › Queue › User Result

User Result

OPTIONS GET

```
GET /api/queue/task/92e223d1-c2d8-4dd0-b2f0-bf739cd2da64/
```

```
HTTP 200 OK
Content-Type: application/json
Vary: Accept
Allow: GET, HEAD, OPTIONS

{
    "task_id": "92e223d1-c2d8-4dd0-b2f0-bf739cd2da64",
    "tags": [],
    "timestamp": "2021-08-31T17:39:55.018",
    "args": [
        {
            "test1": "Hello ",
            "test2": "World"
        }
    ],
    "queue": "celery",
    "user": "ecopad",
    "task_name": "ecopadq.tasks.tasks.test",
    "kwargs": {},
    "result": {
        "status": "SUCCESS",
        "traceback": null,
        "result": "Hello World",
        "date_done": "2021-08-31T17:39:57.216",
        "children": []
    }
}
```

If there's anything wrong, you will be notified in this page. You may fix the bug with the information provided in this page.