RIA-653549

OneData demo

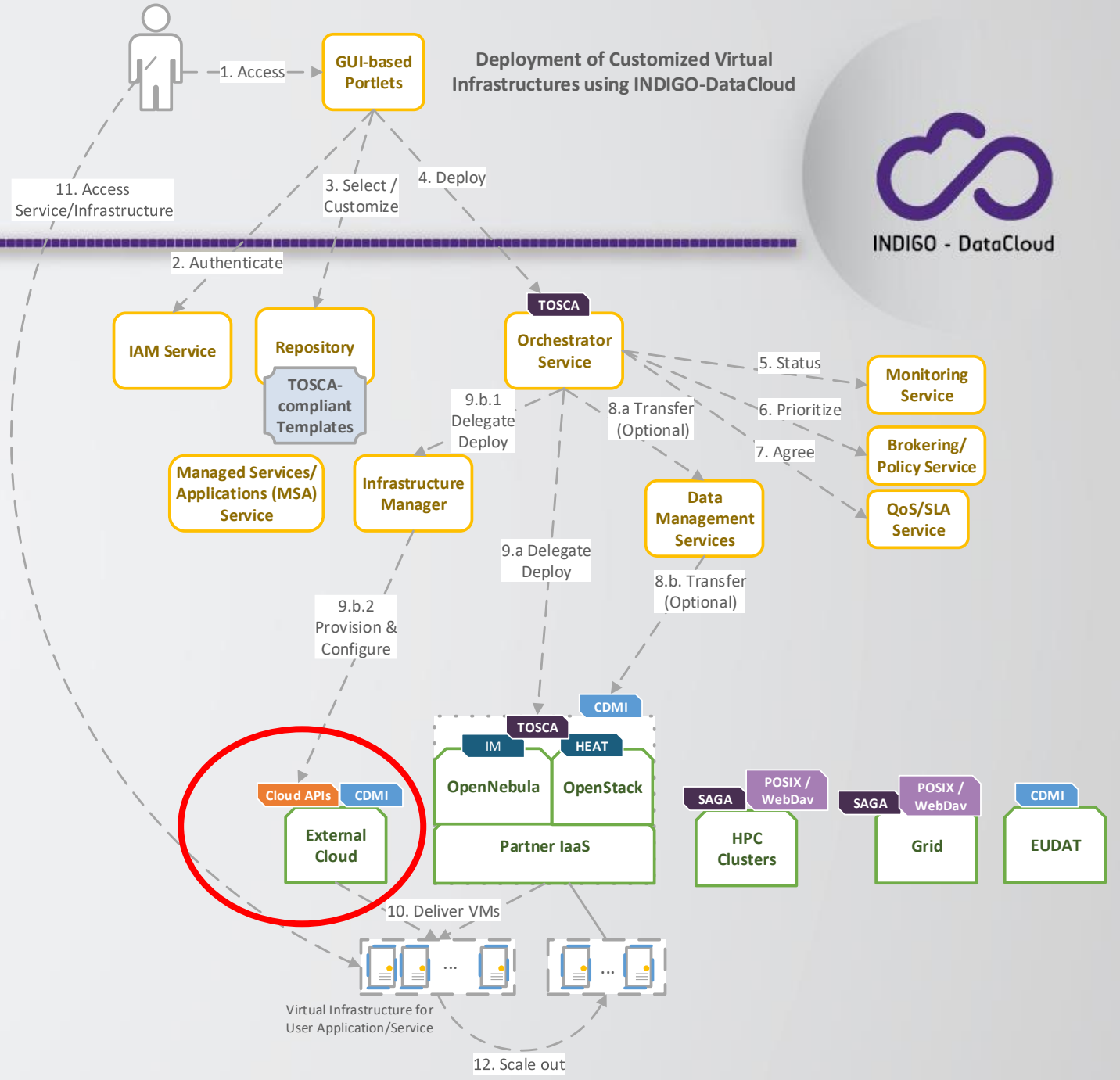
Alfonso Pérez, Germán Moltó

Universitat Politècnica de Valènica

INDIGO-DataCloud is co-founded by the Horizon 2020Framework Programme

Goals

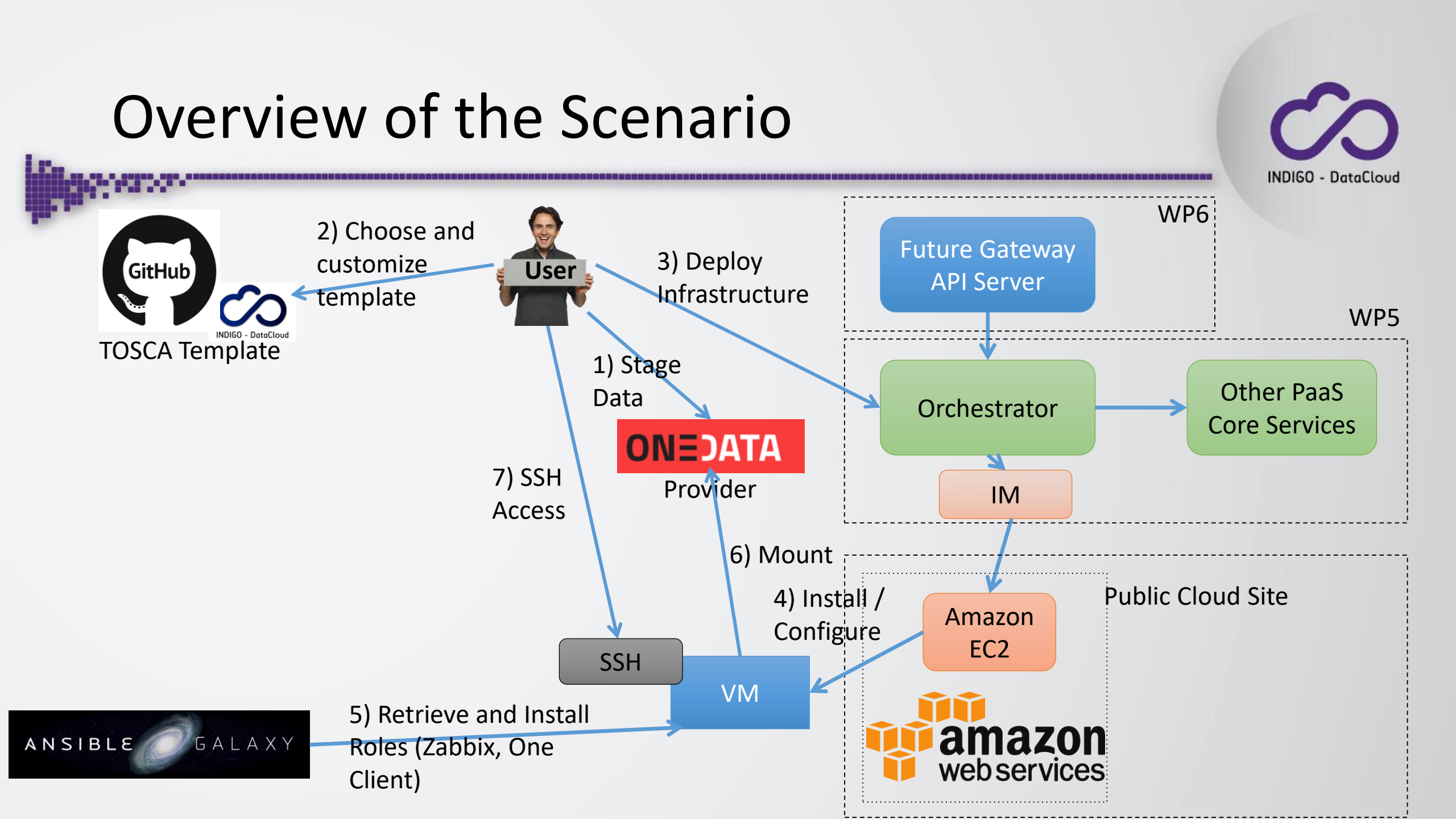
- Introduce support to deploy infrastructures described on TOSCA templates on public Clouds (e.g. Amazon Web Services, Microsoft Azure) across the whole execution chain in the PaaS layer.
- Integrate with OneData to access data spaces from the virtual infrastructure.



A Demo is Worth a Thousand Words

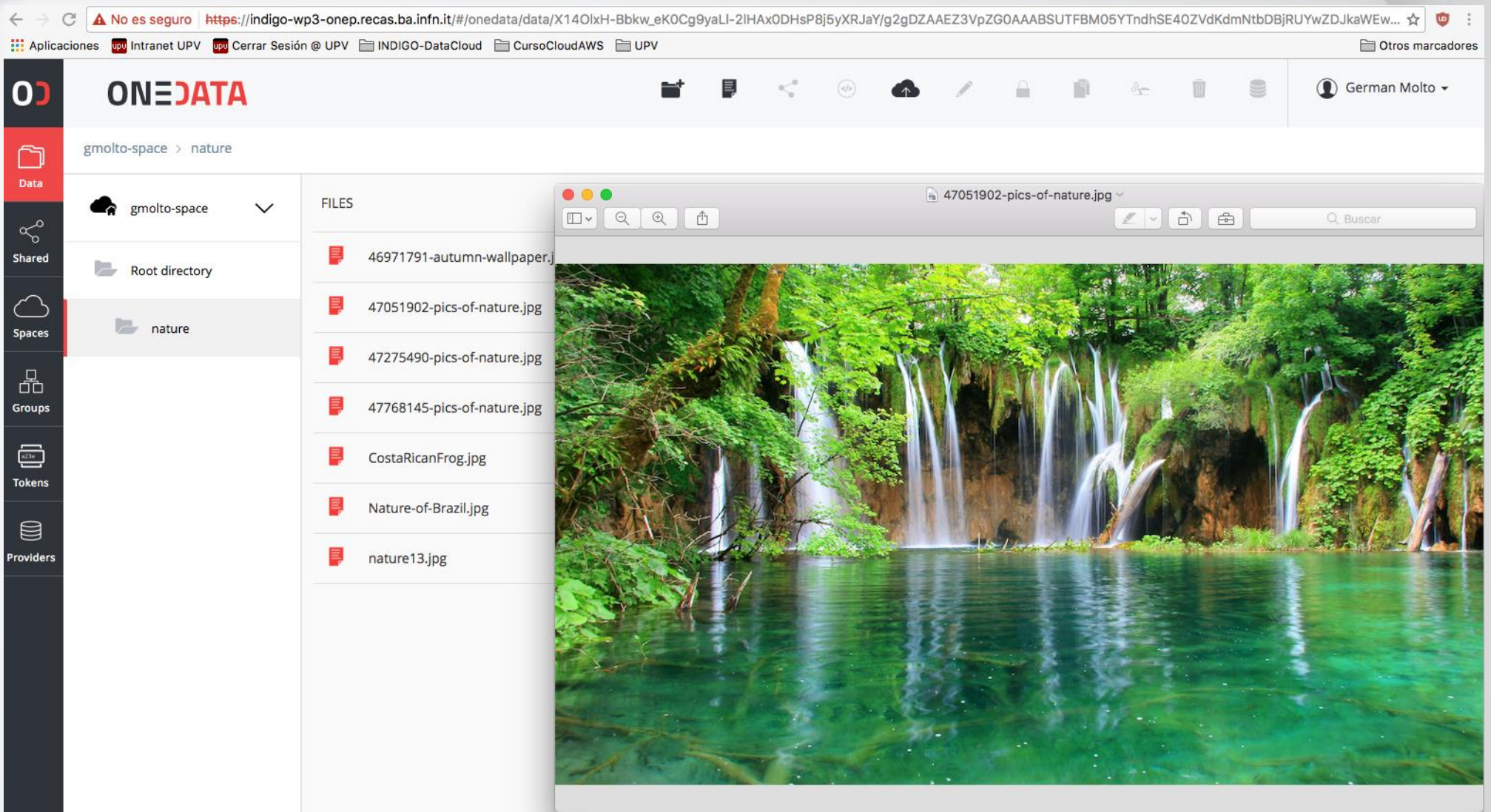
- You have a dataset of images that you want to convert to black & white and the following restrictions:
  - The set of images is stored in a OneData space (and the converted images must be in that OneData space as well).
  - You have an account at IAM (https://iam-test.indigo-datacloud.eu).
  - You want to deploy the required infrastructure in a public Cloud (in this case: AWS).
  - Your application to convert the images is distributed as a Docker container, stored in Docker Hub.

Overview of the Scenario



Demo (0): Dataset in OneData Space

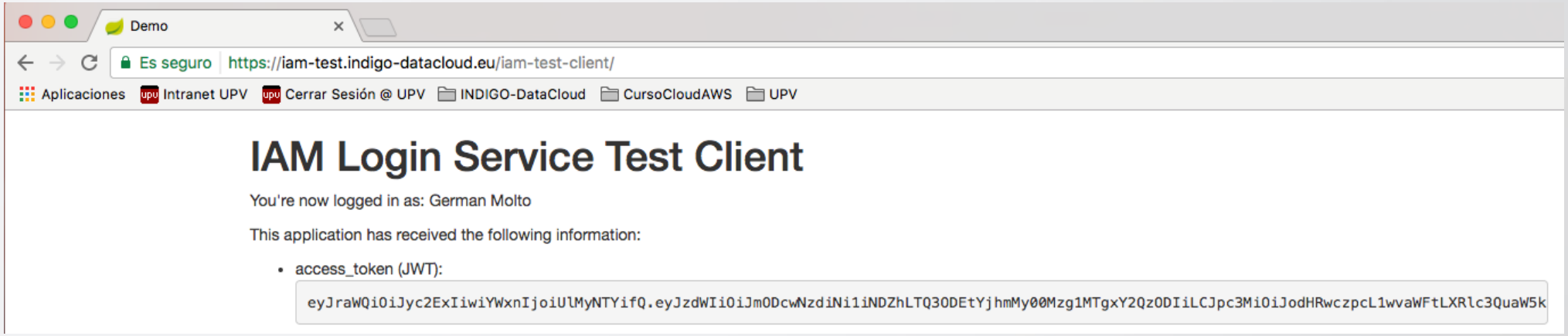
INDIGO - DataCloud



Demo (1): Obtaining the IAM Token

INDIGO - DataCloud

- CLI-based approach:
  - curl -s -L -d client_id=7873d62e-bf8d-4a1b-51b4-a9e6b7afb172 -d client_secret=ALy1LCRoEQA8tpVuOkEDVIr0cNNZecdNCiJ2PKA4HUvmCqyfKlqIQGg8C21Mh1tPgyhH1v98YVdQTOx2JaYf1gw -d grant_type=password -d username=indigo-user -d password=M6dPPnf0GiJ7Ba -d scope="openid address phone profile offline_access email" https://iam-test.indigo-datacloud.eu/token
- Web-based approach:



Demo (2): The TOSCA Template (I)

INDIGO - DataCloud

- Template can be found here: https://github.com/indigo-dc/tosca-types/blob/master/examples/onedata_client_on_aws.yaml

```
22  node_templates:
23    onedata_client_node:
24      type: tosca.nodes.indigo.Compute
25      capabilities:
26        endpoint:
27          properties:
28            network_name: { get_input: network_name}
29            ports:
30              oneclient_5555:
31                protocol: tcp
32                source: 5555
33              oneclient_443:
34                protocol: tcp
35                source: 443
36        scalable:
37          properties:
38            count: 1
```

Specify the subnet of the *VPC* on which the VMs will be provisioned and if the network will be PUBLIC (i.e., the VM will receive a public IP)

Ports that will be open in the *security group* to be accesible from 0.0.0.0/0 (anywhere)

Demo (2): The TOSCA Template (II)

INDIGO - DataCloud

```
host:
  properties:
    num_cpus: 1
    mem_size: 1 GB
os:
  properties:
    type: linux
    distribution: ubuntu
    version: 16.04
    image: us-east-1/ami-3957b02f
    credential:
      user: ubuntu
      token: ''
```

This will be translated to a specific *instance type* in AWS (t2.micro, t2.medium, m3.large, etc.)

The AMI identifier is obtained from the AWS Marketplace or the list of Community AMIs in the EC2 Console. The syntax is specified in: http://imdocs.readthedocs.io/en/devel/radl.html
Notice that the region on which the VM will be deployed is included.
The AMI is based on Ubuntu 16.04

Username to access the VM via SSH (using the private key obtained at deployment time).

Demo (2): The TOSCA Template (III)

INDIGO - DataCloud

```
requirements:
  - local_storage:
      node: my_onedata_storage
      capability: tosca.capabilities.Attachment
      relationship:
        type: tosca.relationships.AttachesTo
        properties:
          location: /tmp/onedataspace
        interfaces:
          Configure:
            pre_configure_source:
              implementation: https://raw.githubusercontent.com/indigo-dc/tosca-types/master/artifacts/onedata/oneclient_install.yml
              inputs:
                onedata_location: { get_property: [ SELF, location ] }
                onedata_token: { get_property: [ TARGET, credential, token ] }
                oneprovider_host: { get_property: [ TARGET, oneprovider_host ] }
                dataspace: { get_property: [ TARGET, dataspace ] }
                onezone_endpoint: { get_property: [ TARGET, onezone_endpoint ] }
```

Local folder of the VM on which the onedata space will be mounted.

Demo (2): The TOSCA Template (IV)

```
my_onedata_storage:
  type: tosca.nodes.indigo.OneDataStorage
  properties:
    oneprovider_host: ['oneprovider.cloud.cnaf.infn.it']
    dataspace: ['alpegon-space']
    onezone_endpoint: https://onezone.cloud.cnaf.infn.it
    credential:
      token: MDAxNWxvY2F00aW9uIG9uZXpvbmUKMDAzYmlkZW500aWZpZXIgM1R1bVZ5dWNBckdwU
      token_type: token
```

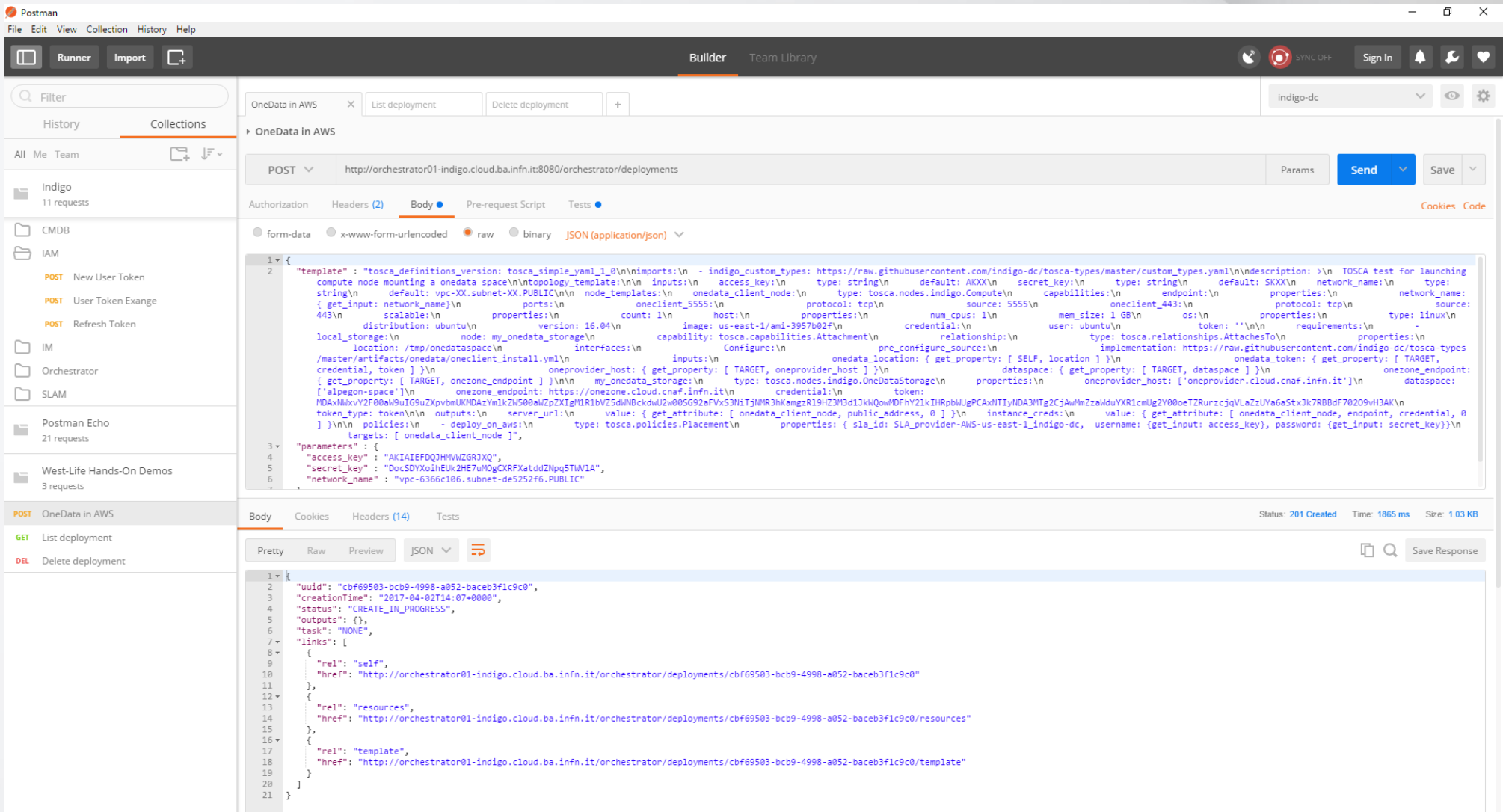
OneData provider supporting the space, the name of the OneData space and the OneZone endpoint.

Access Token required to mount the OneData space in the VM

Demo (3). Deploy the Infrastructure (I)

INDIGO - DataCloud

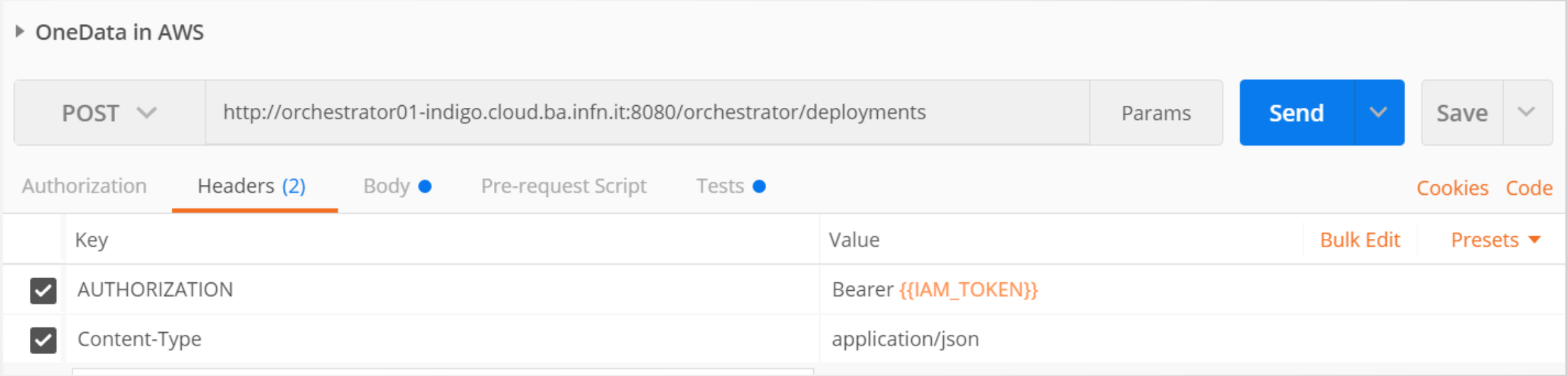
- Postman eases the interaction with the REST API provided by the Orchestrator (and other PaaS Core Services as well)



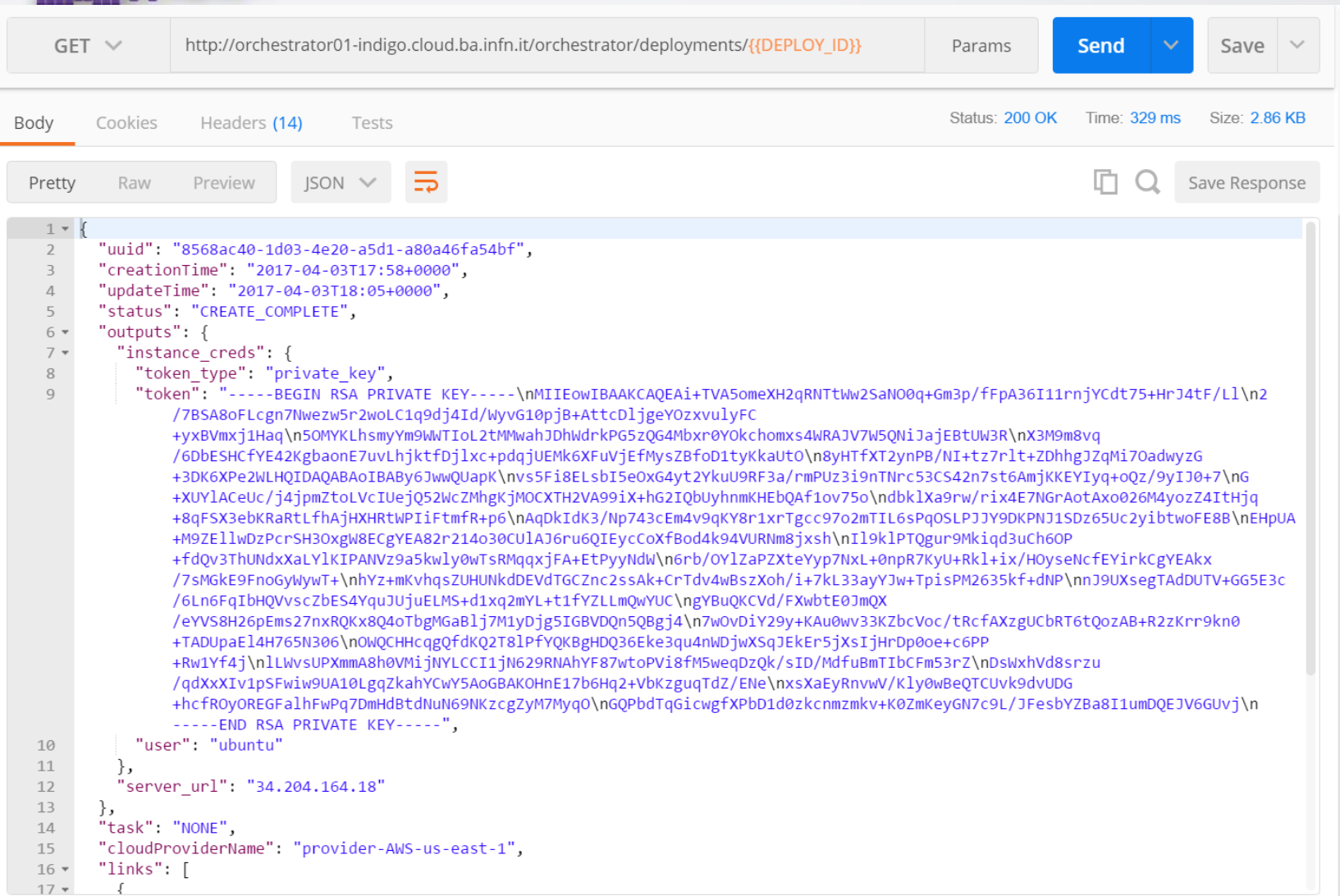
Demo (3). Deploy the Infrastructure (II)

INDIGO - DataCloud

- Headers for the POST Request:
  - AUTHORIZATION: Bearer {{IAM_TOKEN}}
  - Content-Type: application/json



Demo (4). Status of the Infrastructure (I)



- The IM creates an Amazon EC2 keypair and provides you with the corresponding private key used to connect via SSH to the instance.
- State:
  - Running
  - Configured → OK
  - Unconfigured → FAIL (check logs)

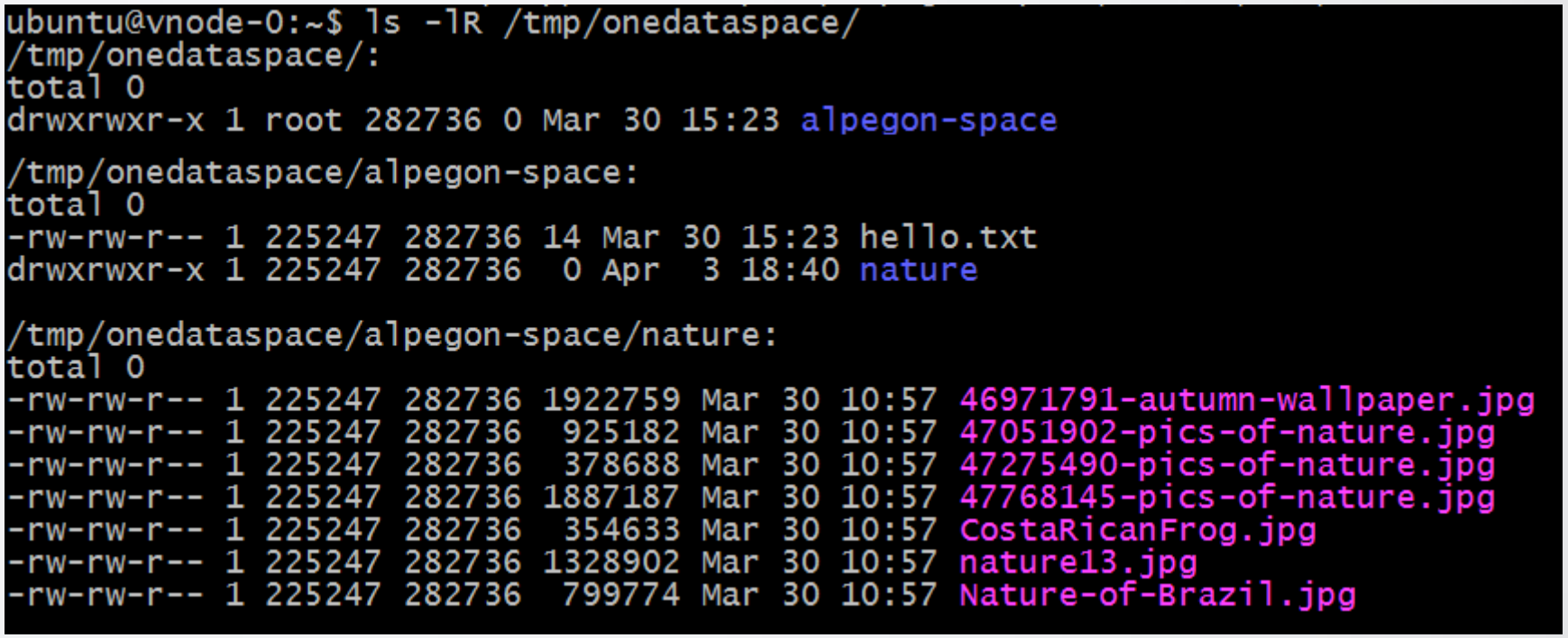
Demo (5). Access the Infrastructure (II)

- Create ssh key:
  - Using Postman scripts
  - Copy value in file (e.g. key.pem)
  - Set access rights:
    - chmod 0600 key.pem
- Connect to the server:
  - ssh -i key.pem ubuntu@34.204.164.18

```
var rawKey = jsonData.outputs.instance_creds.token
if (typeof rawKey !== 'undefined') {
    var parsedKey = rawKey.replace("\\n", "\n");
    postman.setEnvironmentVariable("KEY", parsedKey);
}
```

```
$ ssh -i key.pem ubuntu@34.204.164.18
The authenticity of host '34.204.164.18 (34.204.164.18)' can't be established.
ECDSA key fingerprint is SHA256:zrePk3+uC8mZlg/GN+zUOTD4ZsSEunSHtWxwhuwtB28.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '34.204.164.18' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-53-generic x86_64)
```

Demo (6). Check the OneData space



- OneData space automatically mounted on the VM running on AWS.
- No frills so far, on Ubuntu 16.04
- On Ubuntu 14.04, oneclient gives a segmentation fault.

Demo (7): Mount the OneData space with uDocker.

- We need to use a docker container, but the AMI doens't come with docker intalled
- Can we solve this without requiring root privileges?
  - Yes, with uDocker! Kudos to Jorge Gomes and his team at LIP

```
curl https://raw.githubusercontent.com/indigo-dc/udocker/devel/udocker.py > udocker && chmod u+x udocker
./udocker
./udocker pull knickers/imagemagick
./udocker run -v /tmp/onedataspace/alpegon-space/nature:/tmp/nature knickers/imagemagick ls -l /tmp/nature

executing: ls
total 0
-rw-rw-r-- 1 629476 1796628 1922759 Jan 30 17:12 46971791-autumn-wallpaper.jpg
…
```

Demo (8): Converting all the Files to B&W

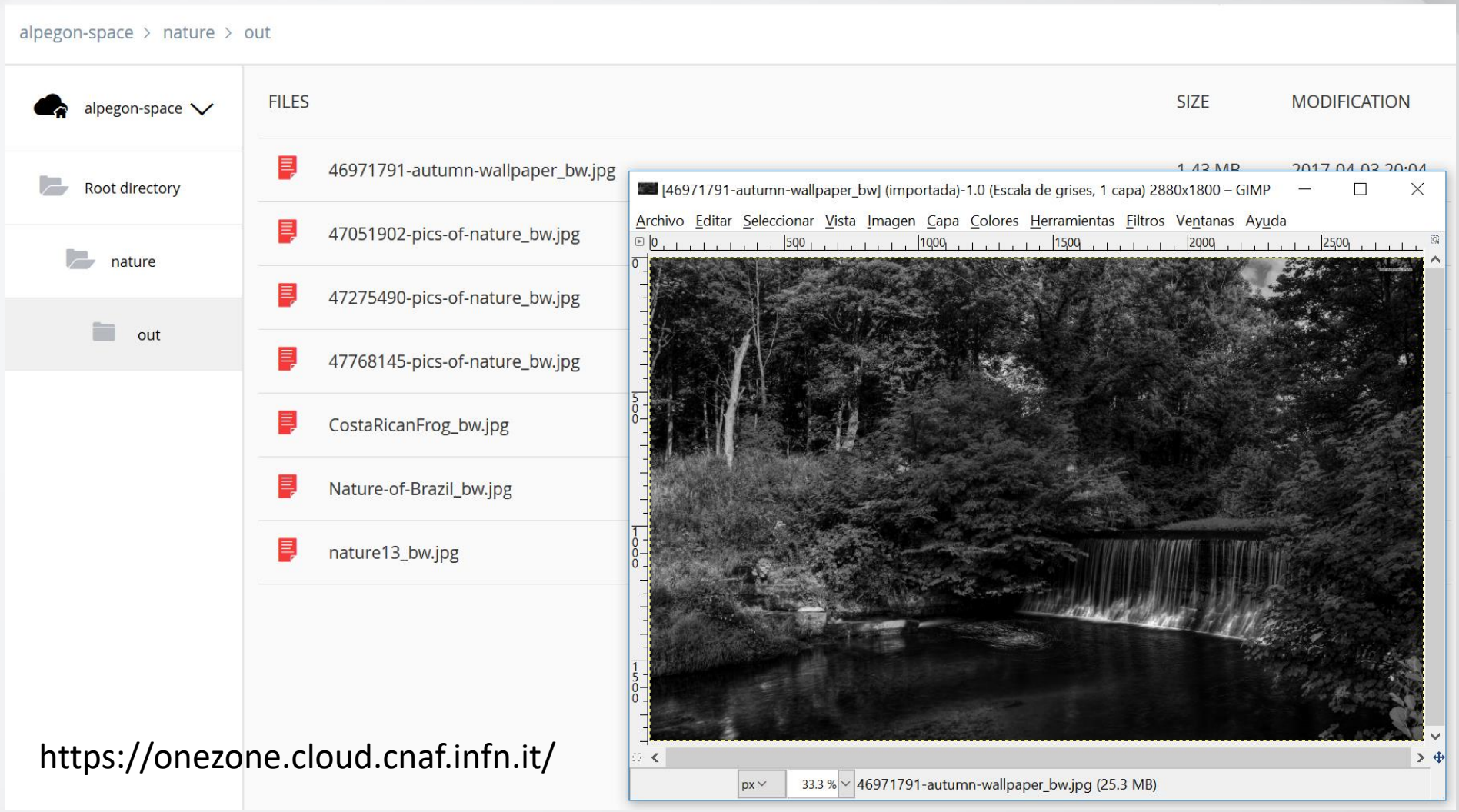

```
./udocker  run --rm -v /tmp/onedataspace/alpegon-
space/nature:/tmp/nature knickers/imagemagick bash

cd /tmp/nature

mkdir -p out; for i in *.jpg; do BASENAME=`basename $i
.jpg`; echo "Converting $i"; convert $i -type Grayscale
out/${BASENAME}_bw.jpg; done
```

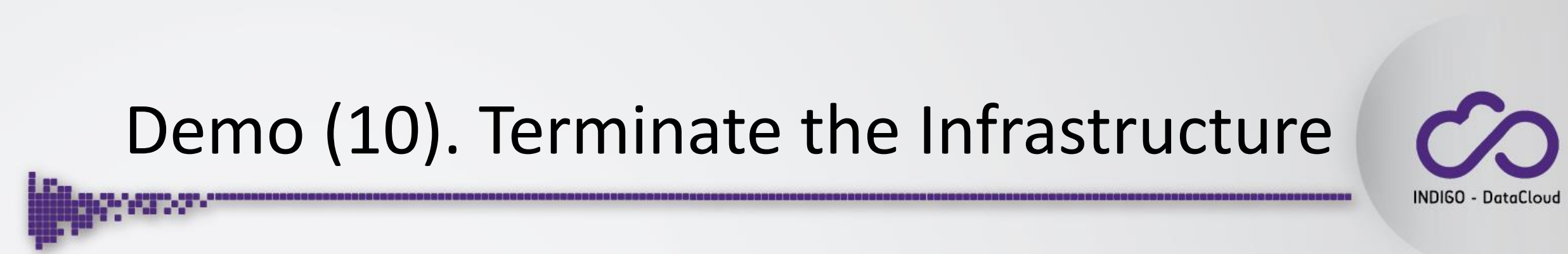
- The volume is mounted inside the Docker container and ImageMagick is employed to batch convert the images.
- Data available in the VM, application running in the Docker container.

Demo (9). Browse the Output Data

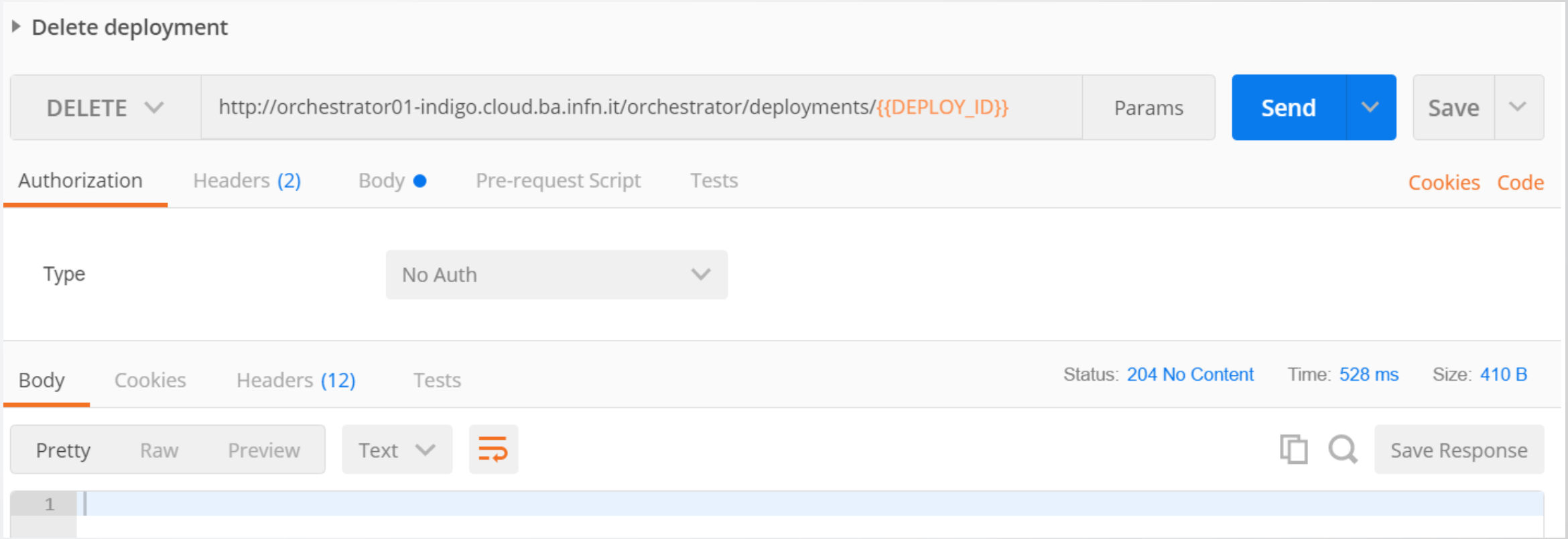


https://onezone.cloud.cnaf.infn.it/

Demo (10). Terminate the Infrastructure



- The resources provisioned in AWS are terminated:
  - EC2 instance, keypair, security group.



Recap of the Demo

INDIGO - DataCloud

- You had:
  - Account on the IAM service
  - AWS credentials (Access Key and Secret Key Id)
  - A OneData space supported by at least one provider.
- During the demo we:
  - Self-provisioned the virtual infrastructure on AWS automatically configured to support Docker (based on the AMI) and with the OneData space automatically mounted, out of a TOSCA template.
  - Staged the required application in a Docker container using uDocker.
  - Mounted the OneData space in the Docker container and performed the image conversion.
  - Data were automatically staged out the virtual infrastructure.

Integrating Support Across the Whole Execution Chain in the PaaS Core

- Support from the CMDB
  - http://indigo.cloud.plgrid.pl/cmdb/service/list

Specify SLA and Credentials in TOSCA

- The user provides the credentials in the TOSCA template.
  - Temporary and limited credentials issued by specific services available in the public Cloud (IAM in the case of AWS).

```
policies:
  - deploy_on_aws:
      type: tosca.policies.Placement
      properties: { sla_id: 4401ac5dc8cfbbb73fr0a02575ee53n6,  username:
AKIAINKHILK6SDVIBTBA, password: JXNYMOcWPFDeCEzPbxoubnoRtNtTBD6Hr3ZCxvMl}
      targets: [ db_server ]
```

Conclusion

- INDIGO-DataCloud is on the road to support public Clouds and this functionality will be available for INDIGO-2.
- Slight changes are required in the TOSCA templates, CMDB, SLAM, Orchestrator and IM
- Involved partners
  - REPLY (Orchestrator), UPV (TOSCA / IM) and CYFRONET (CMDB and SLAM)