Supplementary Section 3. Installing and populating BIAFLOWS locally

It is possible to install BIAFLOWS on a local server or a desktop computer. This might be useful to manage images locally or to analyse them with existing or custom BIAFLOWS workflows. The procedure is described below and should take less than 30 minutes using a regular Internet connection (UNIX based system recommended).

Installing a local instance of BIAFLOWS

The procedure described in this section is for Linux Ubuntu but since BIAFLOWS leverage Docker container technologies, it should be possible to install BIAFLOWS on other platforms (not tested). Some specific details related to deployment on Mac OS can be found online.

1/ Install requirement
BIAFLOWS runs in Docker containers, so that the only requirement is to install Docker.
Check official Docker documentation to install Docker for Ubuntu. In particular, choose Install using the repository, set up the repository and install Docker CE.

2/ Retrieve BIAFLOWS installation files
```
mkdir Biaflows/
cd Biaflows/
git clone https://github.com/Neubias-WG5/Biaflows-bootstrap.git
cd Biaflows-bootstrap
```

3/ Configure the local instance
Edit configuration.sh file and if needed, update URLs (CORE_URL, IMS_URL, UPLOAD_URL). Make sure to use URLs that are not already used by other applications (avoid localhost) to prevent conflicts. Add XXX_URL variable values into the **`/etc/hosts`** of the host machine. In the **/etc/hosts**, add the following lines and don't forget to adapt them with values previously chosen for XXX_URL variables in the configuration.sh file.

```
127.0.0.1    biaflows
127.0.0.1    biaflows-ims
127.0.0.1    biaflows-upload
127.0.0.1    rabbitmq
```

If needed, update data path variables (IMS_STORAGE_PATH …). All data paths must be valid and mappable in the Docker engine. Create all the directories (**mkdir**) corresponding to the following variables (if they don't exist):
IMS_STORAGE_PATH
IMS_BUFFER_PATH
FAST_DATA_PATH
PROXY_CACHE_PATH

SOFTWARE_CODE_PATH
SOFTWARE_DOCKER_IMAGES_PATH
JOBS_PATH
SERVER_SSHKEYS_PATH

Configure `BIAFLOWS_WORKFLOWS_METRICS` to *true* or *false* depending if you want to perform benchmarking (ground truth annotations are then required for all images), or only plan to manage and process images with BIA workflows.

4/ Initialize the deployment

Generate the installation script with the command:

```
sudo bash init.sh
```

5/ Deploy the local instance

Run the generated deployment script with the command

```
sudo bash start.sh
```

6/ Check the running instance

When start up is finished, check the application is running in your browser on the URL specified in your CORE_URL variable (by default: http://biaflows).
Three accounts are created by default (username: admin; password: admin; username: guest; password: guest; username: neubias; password: neubias) with different access rights. Passwords should be updated from the Account page at the top right.

7/ Install sample projects (images and ground-truth data)

After BIAFLOWS successfully installed locally, this local instance is still empty of data. All projects available on BIAFLOWS maintained by NEUBIAS can be imported to this local instance.

Get the public and private keys of the admin account (at the end of the Account page). Then, run:

```
cd Biaflows-bootstrap
sudo bash ./inject_demo_data.sh ADMIN_PUBLIC_KEY ADMIN_PRIVATE_KEY
```

where `ADMIN_PUBLIC_KEY` and `ADMIN_PRIVATE_KEY` have been substituted by their respective values.

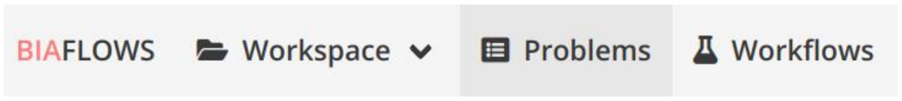
The script starts to download projects and import them in your local BIAFLOWS.
The list of imported projects can be tweaked by editing the file
`Biaflows-bootstrap/configs/project_migrator/projects.txt.`

The whole data injection procedure can take several minutes, depending on your Internet connection and the number of projects being imported.

Creating a new problem (project) in a local BIAFLOWS instance

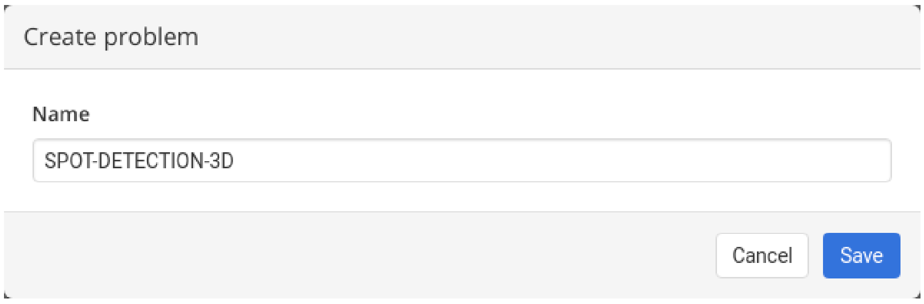
To create a new problem, connect as regular user or admin.

1/ Go to **Problems** tab



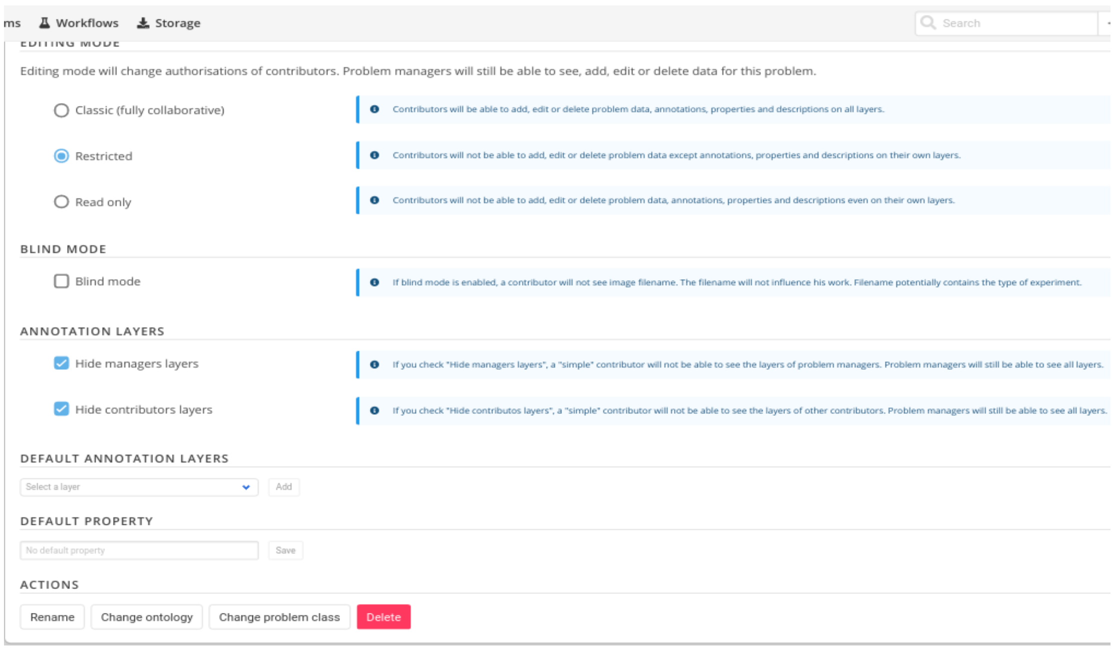
2/ Click **New Problem**

3/ Choose a meaningful problem name and save


Create problem

Name: SPOT-DETECTION-3D

4/ The problem is ready to be configured, the following configuration is recommended



EDITING MODE

Editing mode will change authorisations of contributors. Problem managers will still be able to see, add, edit or delete data for this problem.

- ○ Classic (fully collaborative) — Contributors will be able to add, edit or delete problem data, annotations, properties and descriptions on all layers.
- ● Restricted — Contributors will not be able to add, edit or delete problem data except annotations, properties and descriptions on their own layers.
- ○ Read only — Contributors will not be able to add, edit or delete problem data, annotations, properties and descriptions even on their own layers.

BLIND MODE

- ☐ Blind mode — If blind mode is enabled, a contributor will not see image filename. The filename will not influence his work. Filename potentially contains the type of experiment.

ANNOTATION LAYERS

- ☑ Hide managers layers — If you check "Hide managers layers", a "simple" contributor will not be able to see the layers of problem managers. Problem managers will still be able to see all layers.
- ☑ Hide contributors layers — If you check "Hide contributos layers", a "simple" contributor will not be able to see the layers of other contributors. Problem managers will still be able to see all layers.

DEFAULT ANNOTATION LAYERS

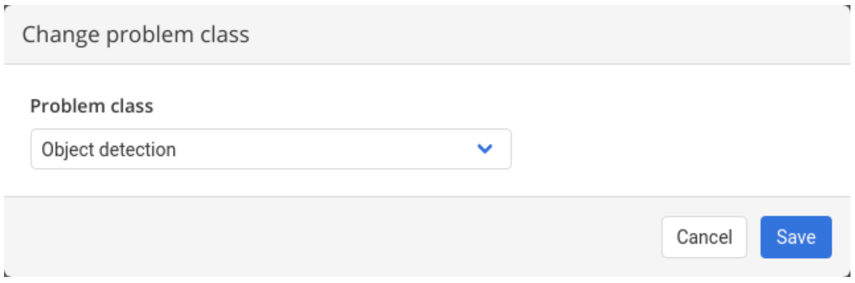
Select a layer

DEFAULT PROPERTY

No default property

ACTIONS

5/ Map your problem to the corresponding problem class (see **Ground truth annotations formats and reported metrics**) by clicking on **Change problem class**. The problem class specifies the format of ground truth annotations (and workflow outputs), as well as the associated benchmark metrics to be computed.



6/ Configure project members. If you work alone, you can leave contributors and project managers to default user. This can be done from the “Members” tab in the problem configuration.

7/ The problem can be fully configured to display or hide panels / tabs / tools in the user interface. This is achieved from the **Custom UI** tab in the problem configuration.

8/ A description of the problem can optionally be added from the **Information** (left sidebar). The description is displayed in **Problems** list.

Uploading images to a local BIAFLOWS instance

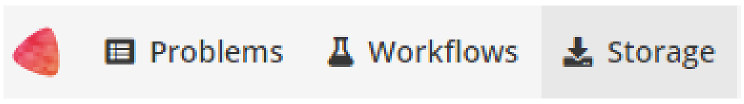
To upload new images, connect as regular user or admin.

<u>Supported formats</u>

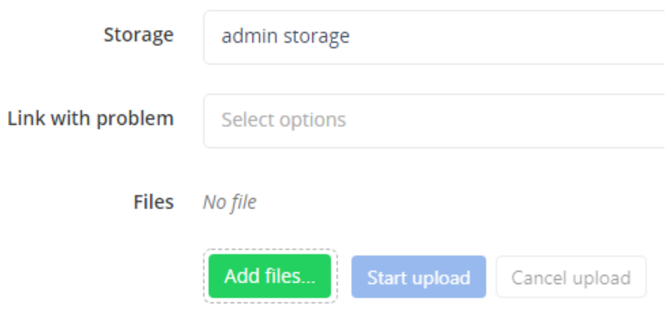
- **2D images**: 8-bit/16-bit TIFF (or OME-TIFF files)
- **Multi-dimensional images (Z, C, T)**: single file 8-bit/16-bit OME-TIFF

Images name should not hold the text string **_lbl** (reserved keyword for ground truth)

1/ Go to **Storage** section



2/ Select the **Problem** to which the images should be associated with (**Link with problem**)



Note: If a problem is not in the list, make sure you are a member for this problem

3/ Click on **Add files…** and select the files from the file browser

4/ Start upload with **Start upload** and wait until completion

The status can be:

- **DEPLOYED/CONVERTED**: The file is correctly imported to BIAFLOWS
- **ERROR FORMAT**: The file format is not supported
- **ERROR EXTRACTION**: Something went wrong during metadata extraction
- **ERROR CONVERSION**: Something went wrong during the conversion of the file into the BIAFLOWS internal image format
- **ERROR DEPLOYMENT**: Something went wrong during the communication with BIAFLOWS API. It can be due to access rights, or other unexpected error

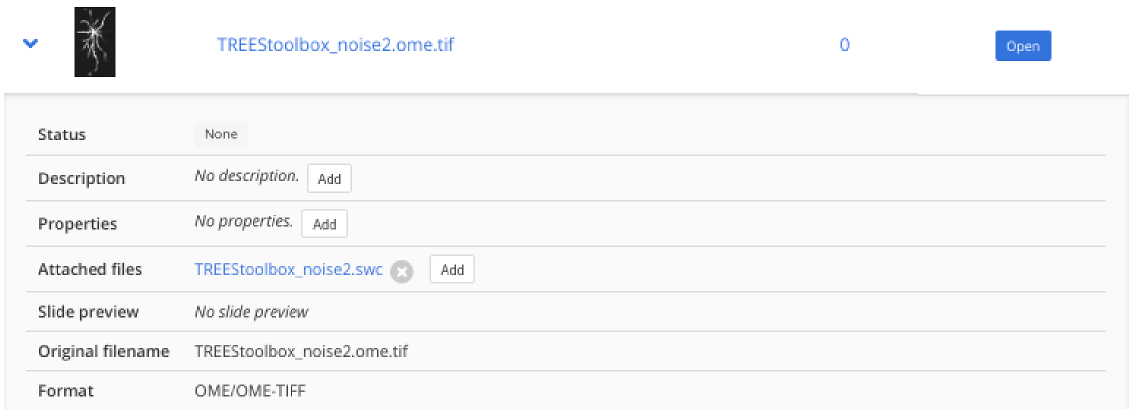
Note: Images uploaded to storage can also be associated to a Problem after upload (Problem: **Add image**). This can be useful to associate the same image to several Problems.

Uploading ground truth annotations to an existing BIAFLOWS problem

If you plan to perform benchmarking, ground truth annotations should also be uploaded and associated to each image of a problem. The format of these annotations depends on the associated problem class (see **Ground truth annotations formats and reported metrics**).

Image annotations (e.g. binary masks) should be uploaded as 16-bit TIFF (or OME-TIFF) for 2D images and single file 16-bit OME-TIFF for multidimensional (C,Z,T) images. They should be uploaded by following the procedure described in the previous section and should have the same name as their corresponding image + **_lbl** suffix (e.g. **AnImage.ome.tif** is the original image and **AnImage_lbl.ome.tif** is the ground-truth).

Other types of annotations (e.g. SWC, text file) should be added to the images as attached files. To do so, expand the image (blue arrow) in the list and click on **Add** next to **Attached files**.



TREEStoolbox_noise2.ome.tif 0 Open

Status	None
Description	*No description.* Add
Properties	*No properties.* Add
Attached files	TREEStoolbox_noise2.swc Add
Slide preview	*No slide preview*
Original filename	TREEStoolbox_noise2.ome.tif
Format	OME/OME-TIFF

Adding existing workflows from trusted sources to a local BIAFLOWS instance

It is possible to integrate existing BIAFLOWS workflows to any BIAFLOWS instance. This operation requires configuring an external trusted source made of:

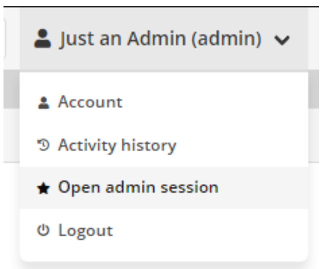
1. A source code registry (typically a Github user space)
2. An execution environment registry (typically a DockerHub user space)

If your workflow repositories are mixed with other repositories in your user space, you can specify a prefix to distinguish workflows repositories. For instance, all bioimage analysis workflows developed by NEUBIAS are prefixed by **W_** and available from this user space: https://github.com/Neubias-WG5.

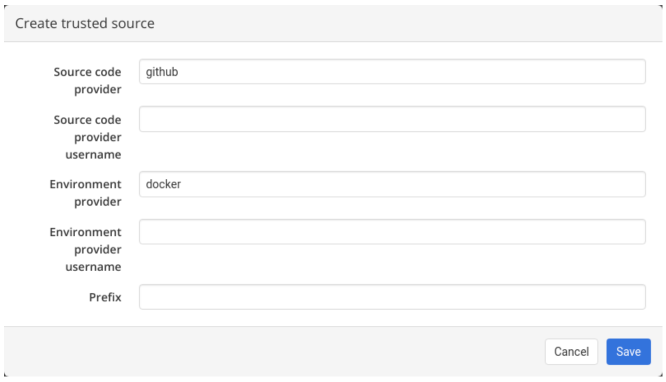
Some information regarding trusted sources is given below.

To manage trusted sources, you need to be administrator.

1/ Connect as administrator by clicking Open admin session:



2/ In administration page, go to **Trusted sources** tab and click **Add trusted source**


Create trusted source

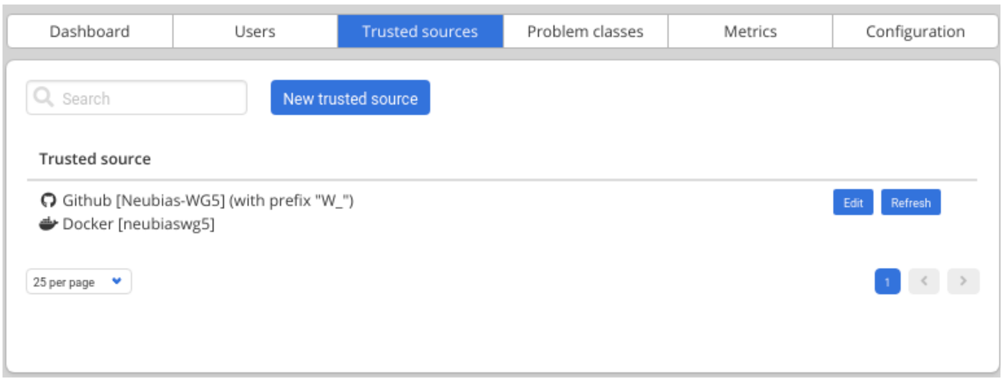
Source code provider	github
Source code provider username	
Environment provider	docker
Environment provider username	
Prefix	

3/ Fill the form and **Save**

For instance, to add NEUBIAS curated set of workflows, the trusted source has to be configured as follows:

- **Source code provider:** github
- **Source code provider username:** Neubias-WG5
- **Environment provider:** docker
- **Environment provider username:** neubiaswg5
- **Prefix:** W_

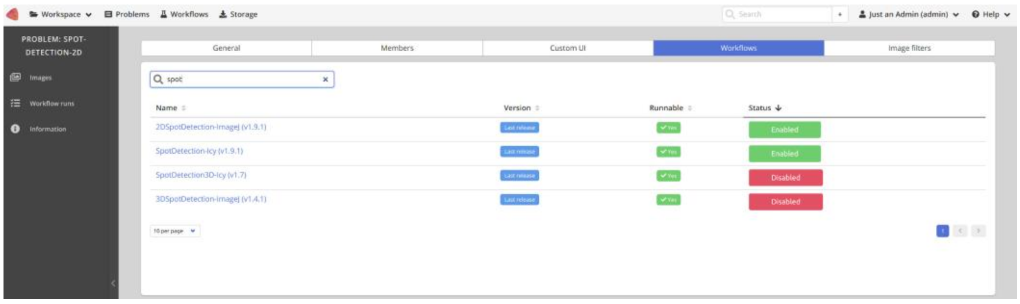
4/ Trusted sources are periodically checked (about every 5/10 minutes) to automatically add new versions of existing workflows or new workflows, but you can also click on **Refresh** to trigger the detection.



Trusted source

Github [Neubias-WG5] (with prefix "W_")
Docker [neubiaswg5]

5) Once a workflow is imported, it has to be linked to a given **Problem** so as to process all the images from this **Problem**. This can be performed in the Configuration panel of the **Problem** (**Workflows** tab) by toggling **Enable** for that workflow as illustrated below:



Name	Version	Runnable	Status
2DSpotDetection-ImageJ (v1.9.1)	Last release	✓ Yes	Enabled
SpotDetection-Icy (v1.9.1)	Last release	✓ Yes	Enabled
SpotDetection3D-Icy (v1.7)	Last release	✓ Yes	Disabled
3DSpotDetection-ImageJ (v1.4.1)	Last release	✓ Yes	Disabled