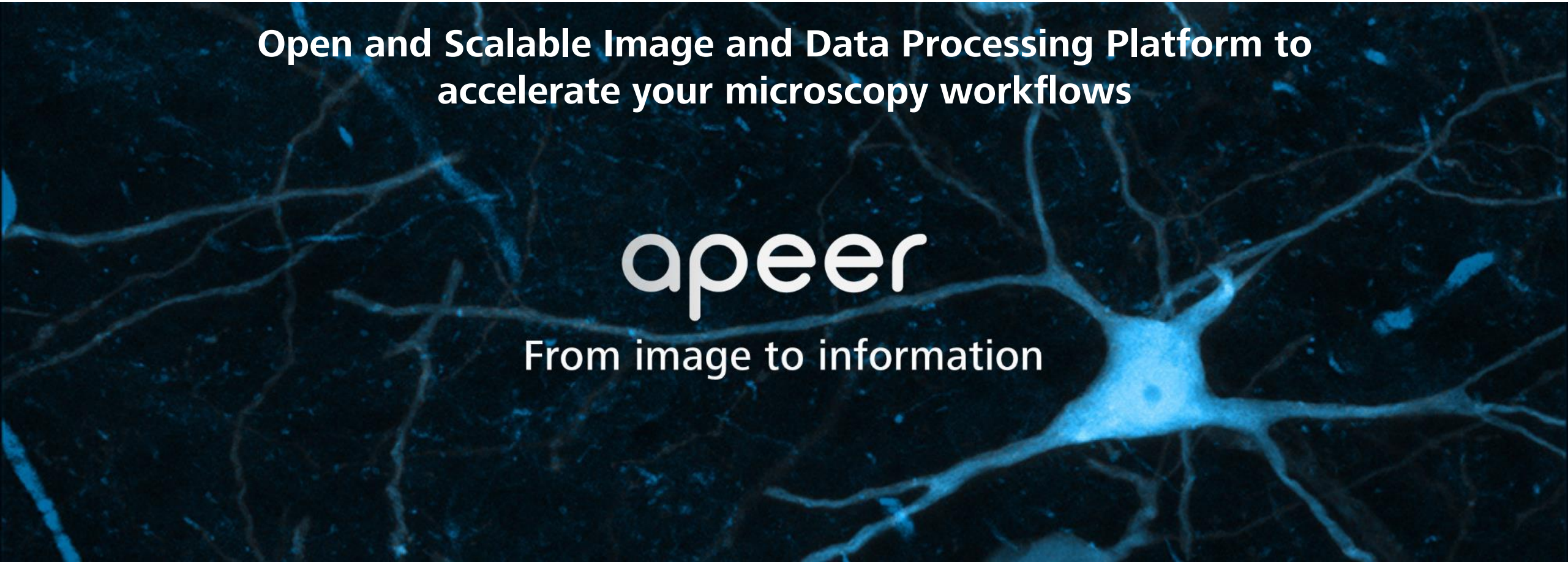
From nanoparticles to neuron segmentation
Accelerating research through digital technology



Dr. Sebastian Rhode

Product Owner, Technology Center Software & Digital Solutions, Munich

The challenges of a microscopist / researcher

Select quotes from user interviews

Researchers with no coding skills cannot use algorithms written by others.

Different students report different results on the same dataset.

If my post doc leaves I'm left with valuable but unusable code.

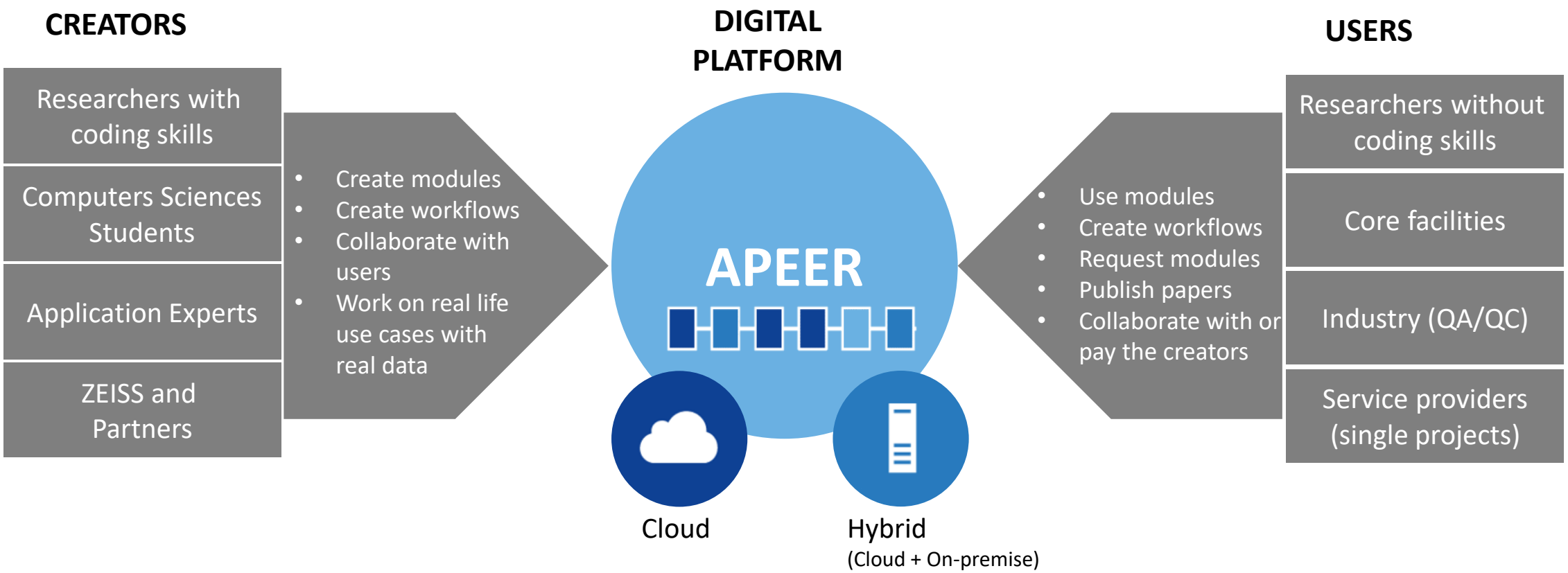
We use different pieces of software with no connectivity between them.

Reproducing published results is a nightmare.

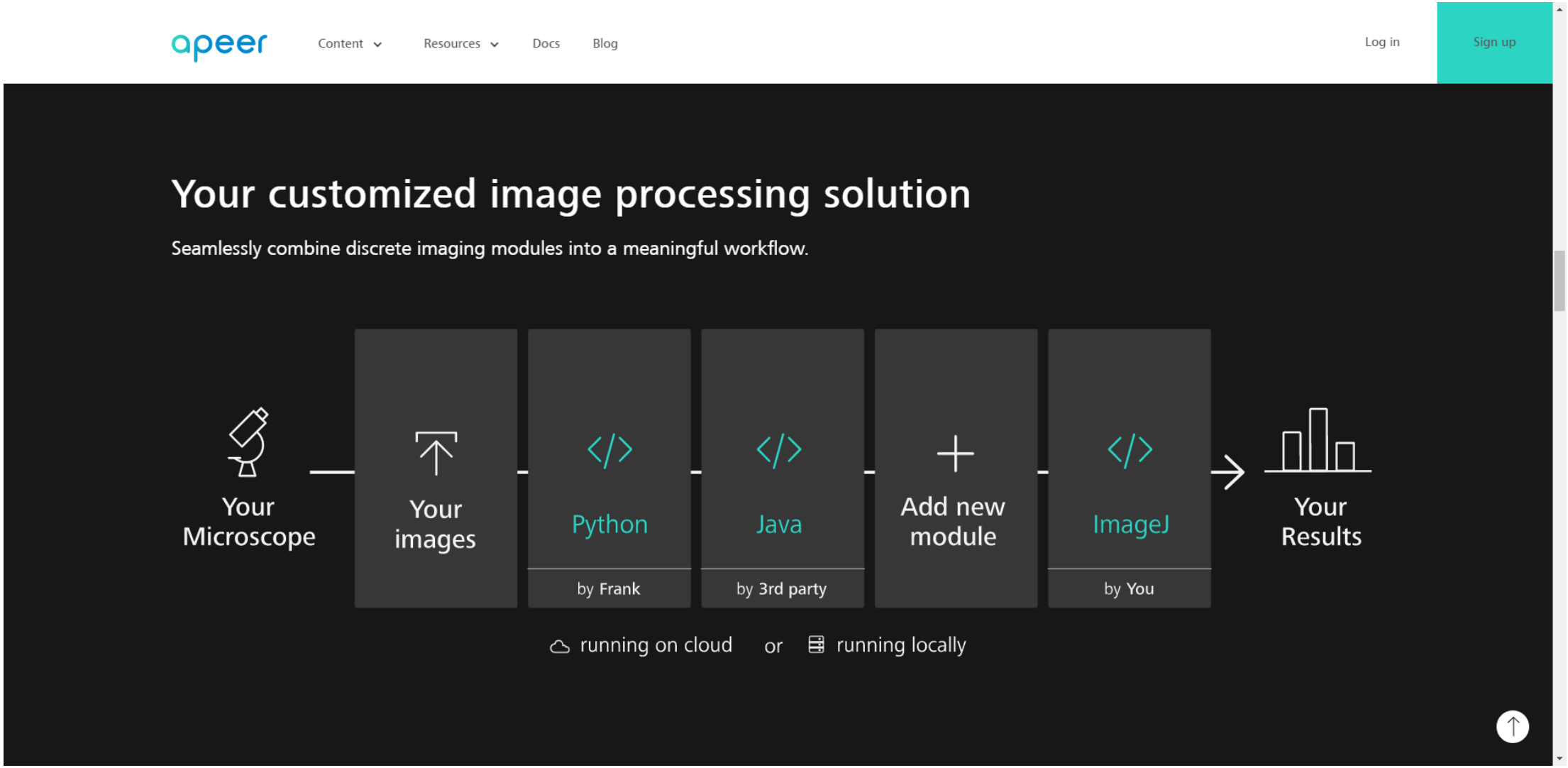
Leveraging digital technologies to accelerate research and improve repeatability

APEER – online platform for microscopists

Goal:
"Drive efficiency in research through a digital platform to easily create, execute and share solutions to various microscopy user challenges."

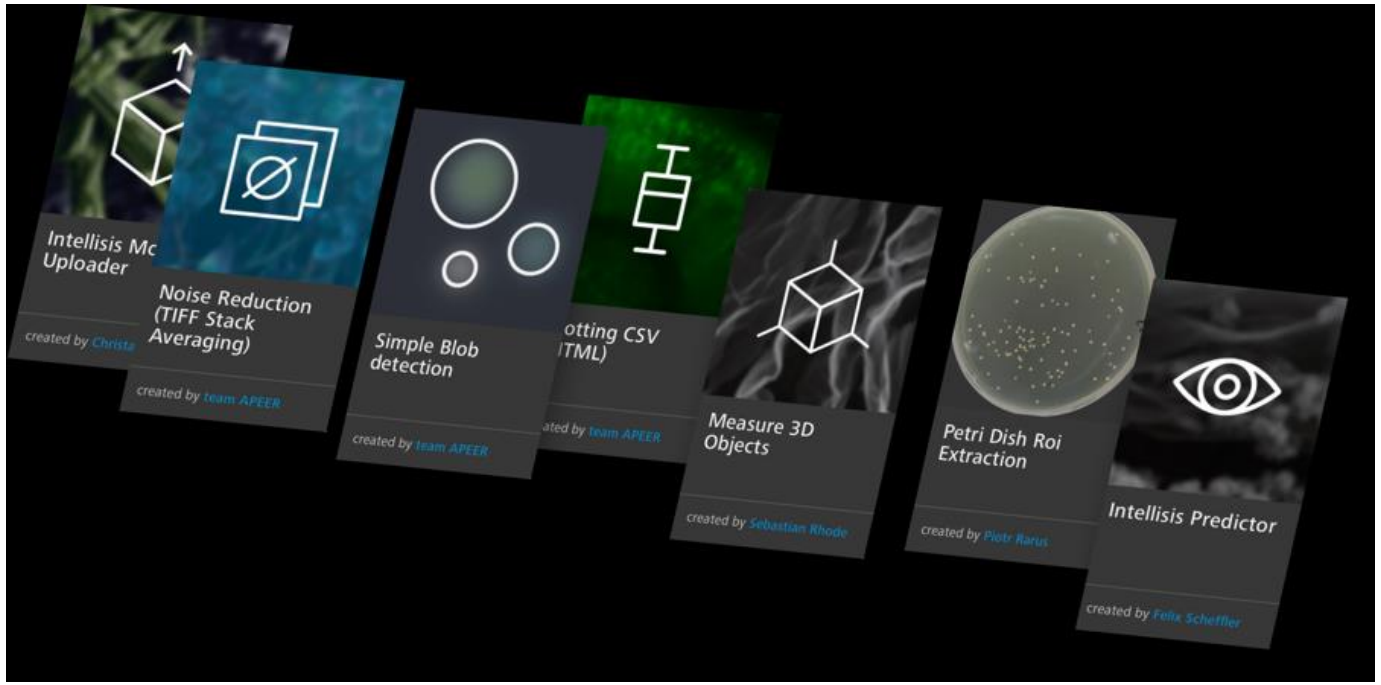


APEER – online platform for microscopists



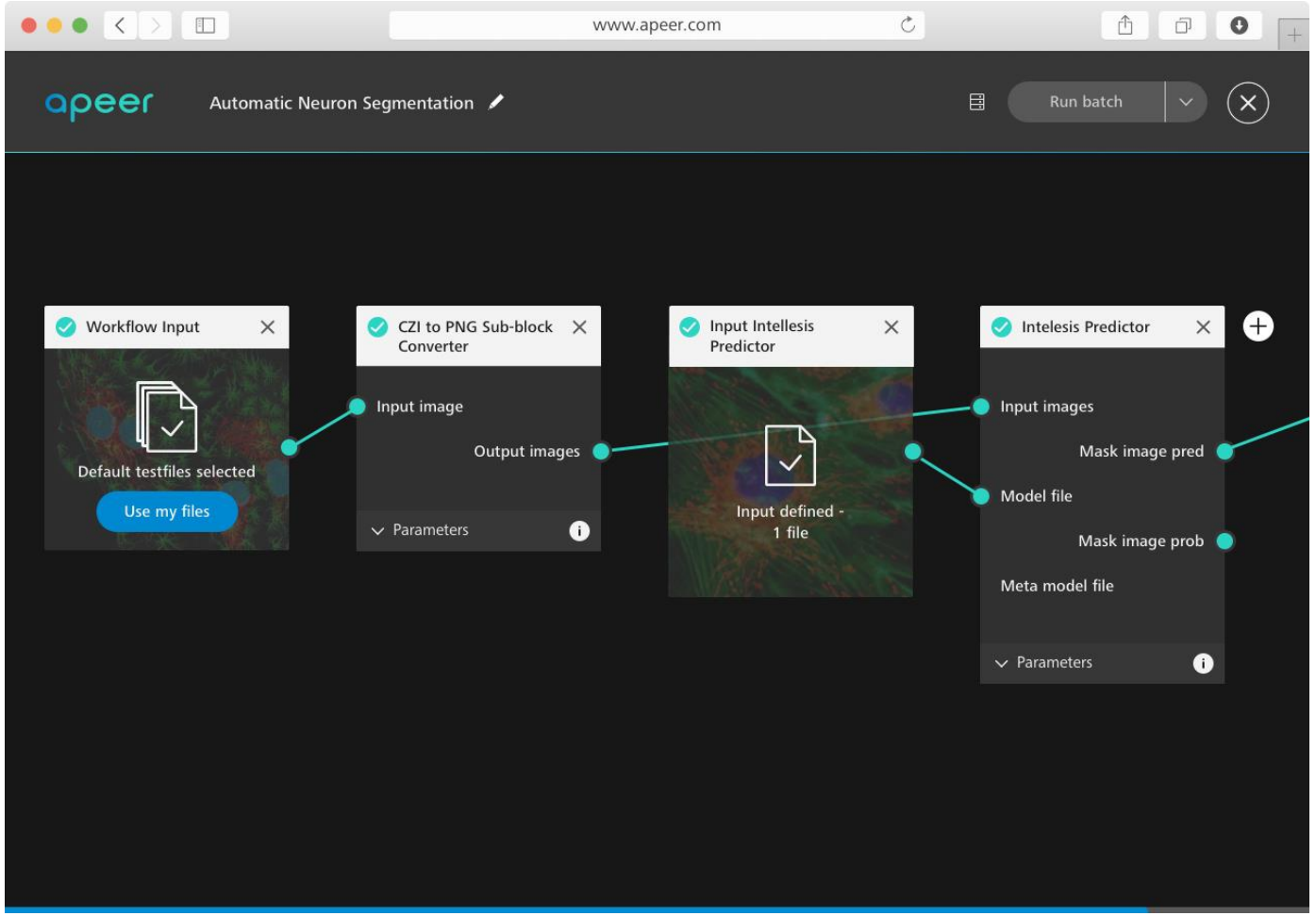
APEER – Create, Combine and Collaborate

Create your own imaging modules using the programming language of your choice, as we use Docker, or build on others' work.



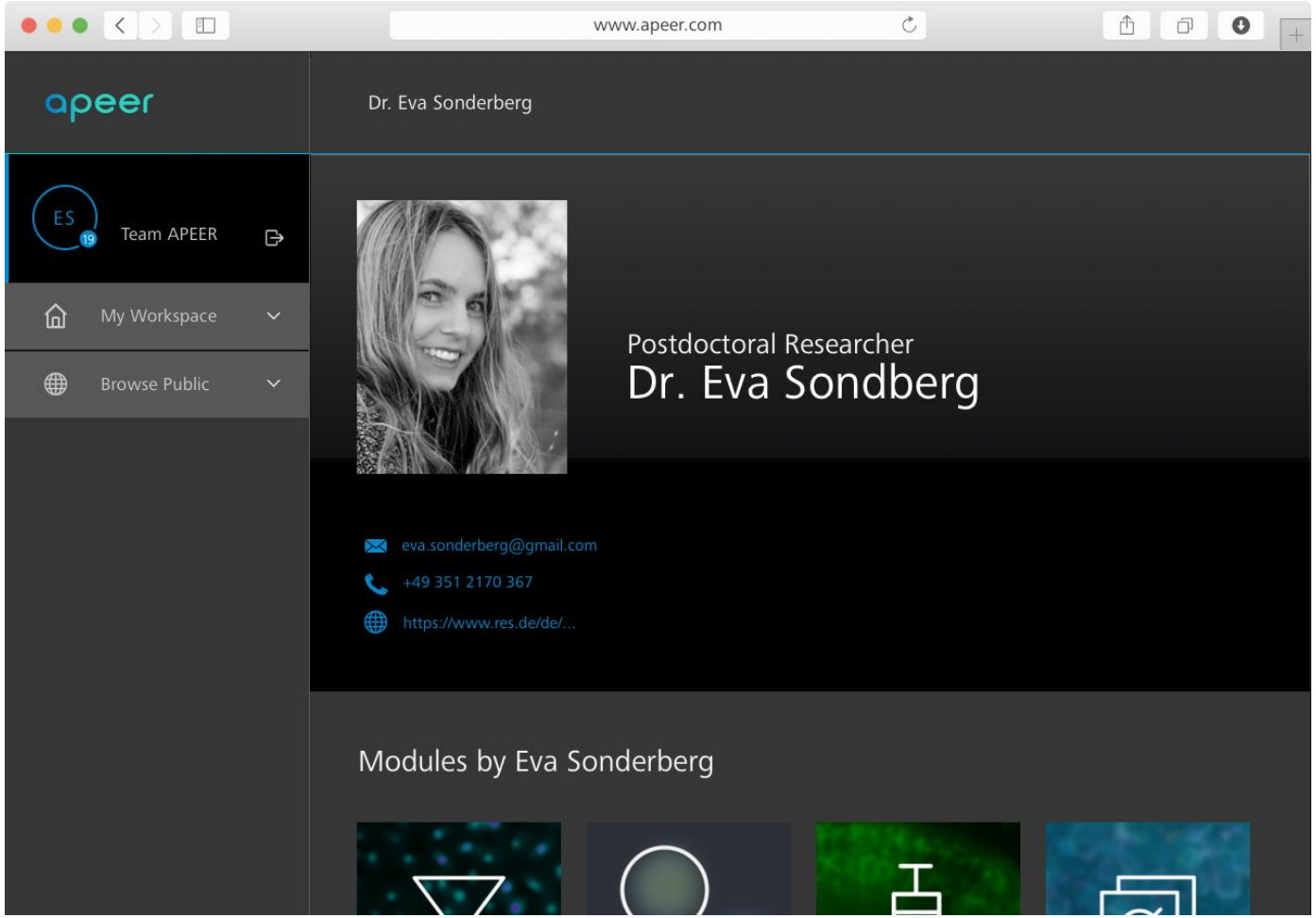
APEER – Create, Combine and Collaborate

Combine private and public image processing modules into customized solutions. Automate the workflows for increased productivity and reproducibility.



APEER – Create, Combine and Collaborate

Collaborate - Save valuable research time by not reinventing code. Share work with your team or with the community. Work from anywhere using any device.



APEER – Encourage reproducible science

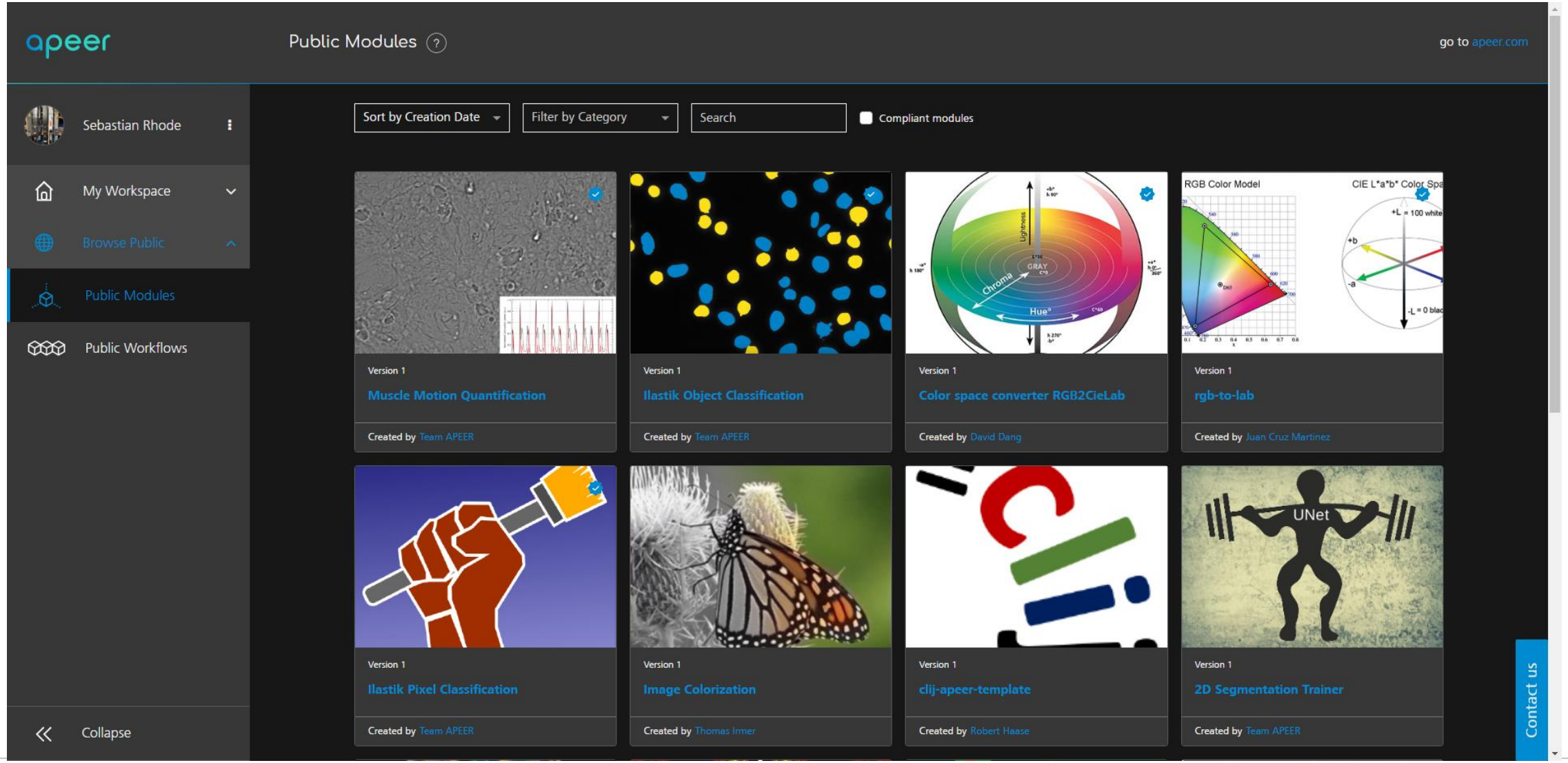
APEER enables you to **create & customize image and data processing solutions for your specific job** to be done.

You can automate tasks by creating modular **workflows** by combining **modules**. Such workflows can be run either in the cloud or locally depending on your preferences.

You can **save time** and conserve knowledge by **using shared modules & workflows**.

By **publishing workflows** together with your results you can encourage other to **reproduce** your findings easily with their data using the exact same software environment with the need to install anything

APEER – Public Modules

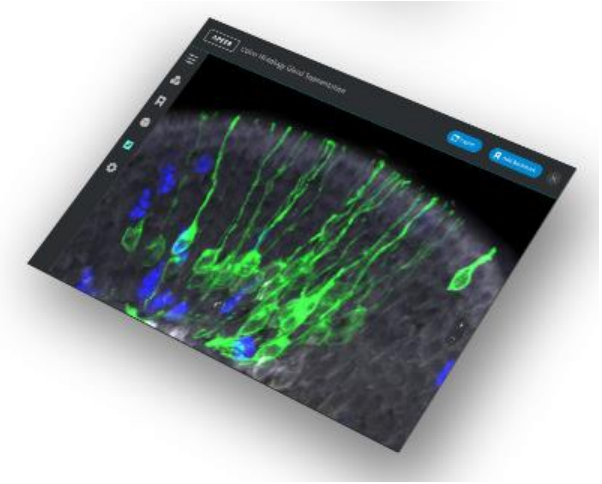


APEER - The journey from image to information

1 Pick modules → 2 Build the workflow → 3 Run the workflow

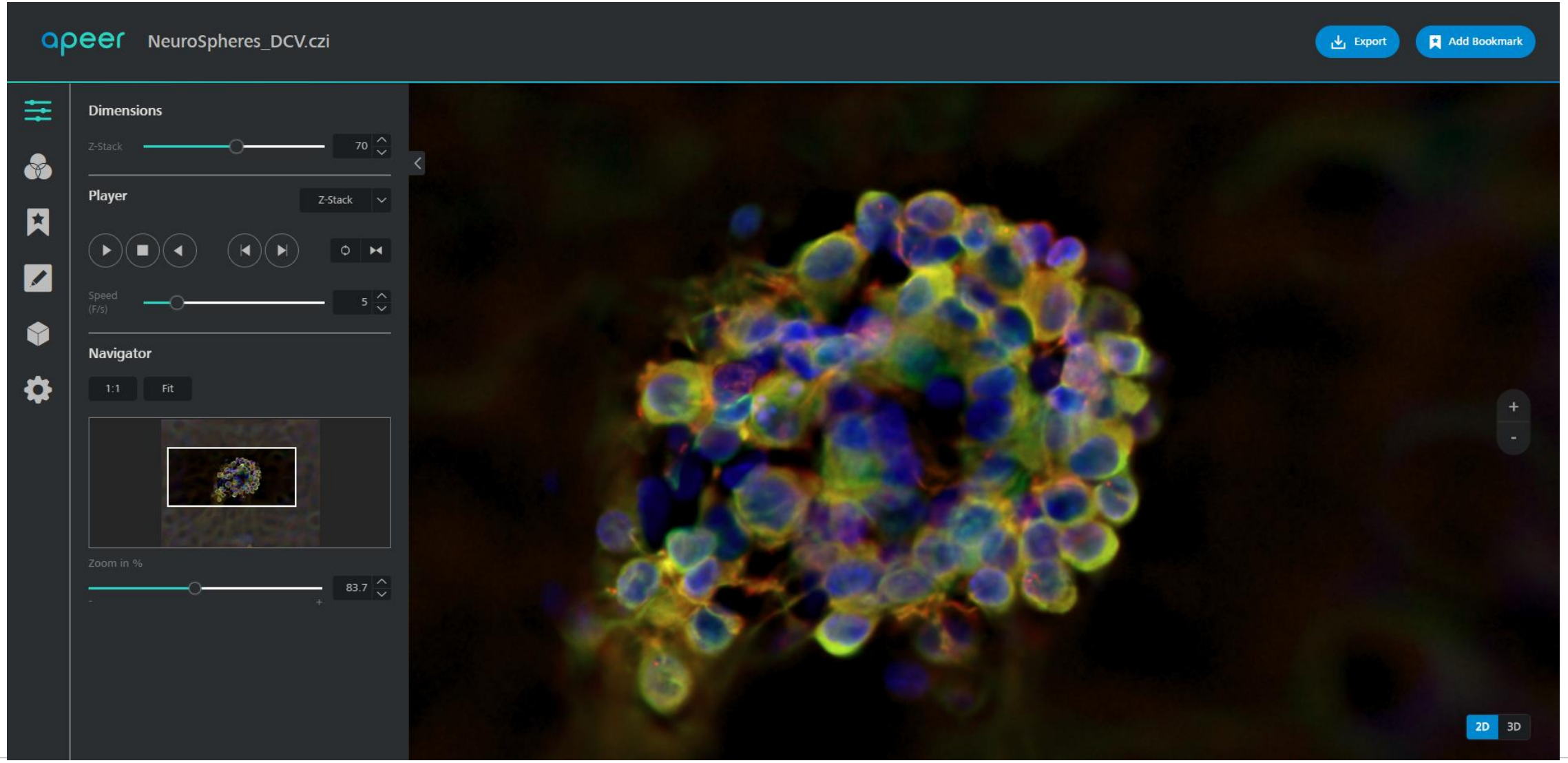
- A module is a piece of software (Docker™) that performs a single task (e.g. file conversion or segmentation)
- Anyone can create a module with a little bit of coding knowledge, **in any language**.

- A workflow is a an arrangement of modules in a specific way customized for a specific job.
- Anyone create a workflow, **no need for coding knowledge**.

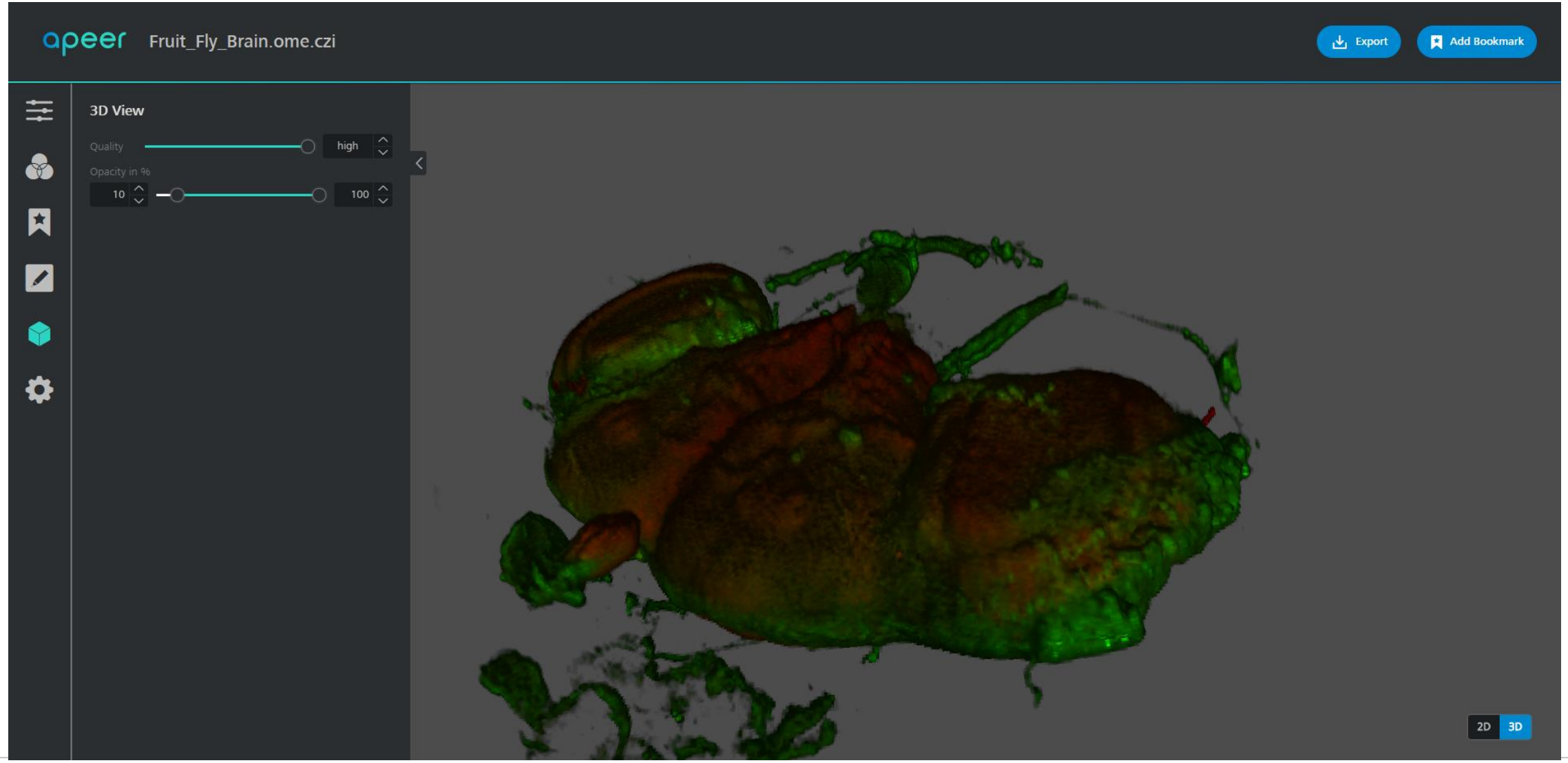


- Run a workflow by just clicking a button.
- Batch runs can be started by selecting multiple input files.
- View and Share the results.

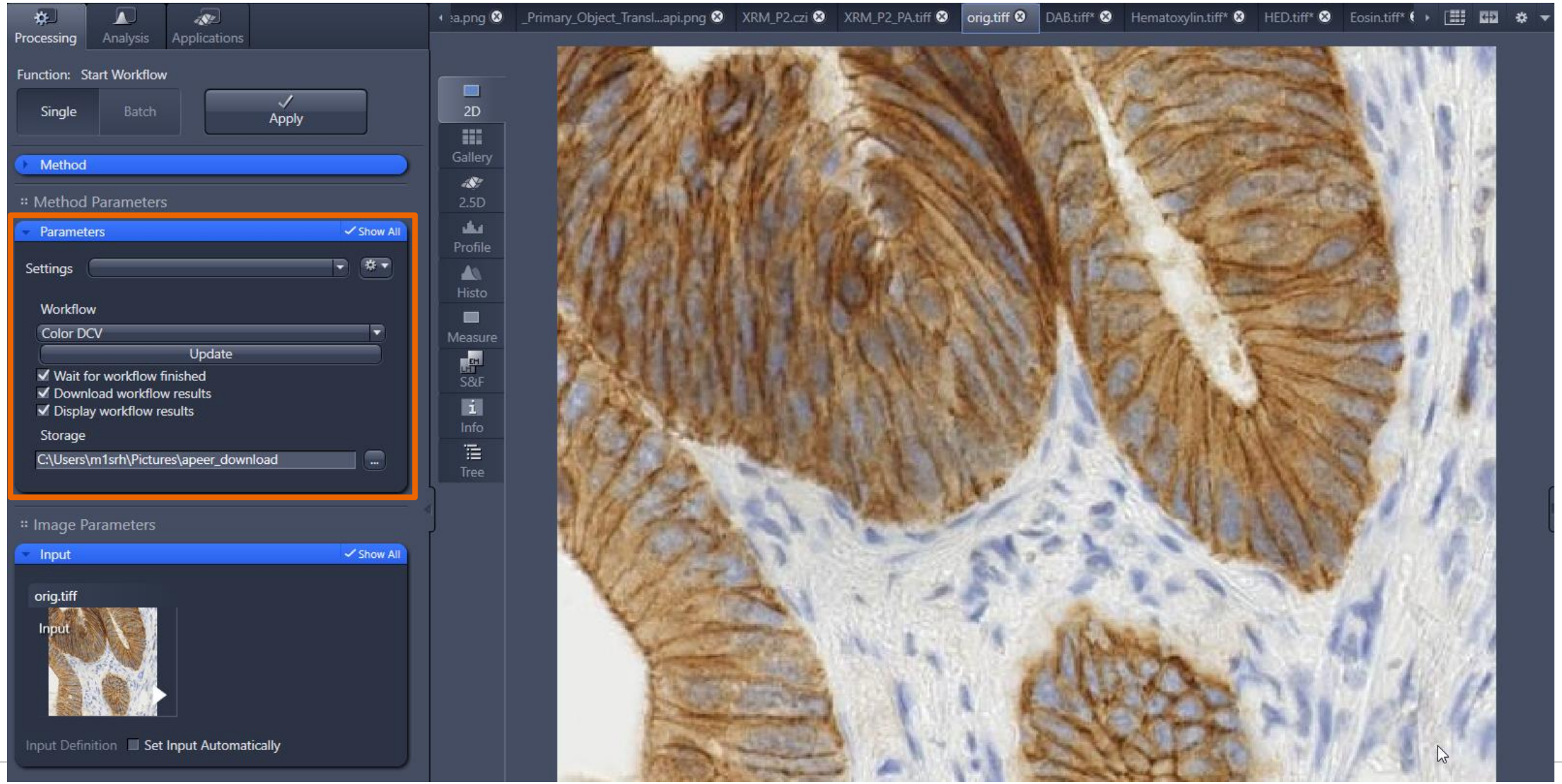
APEER WebViewer - Visualize your images from anywhere using any device



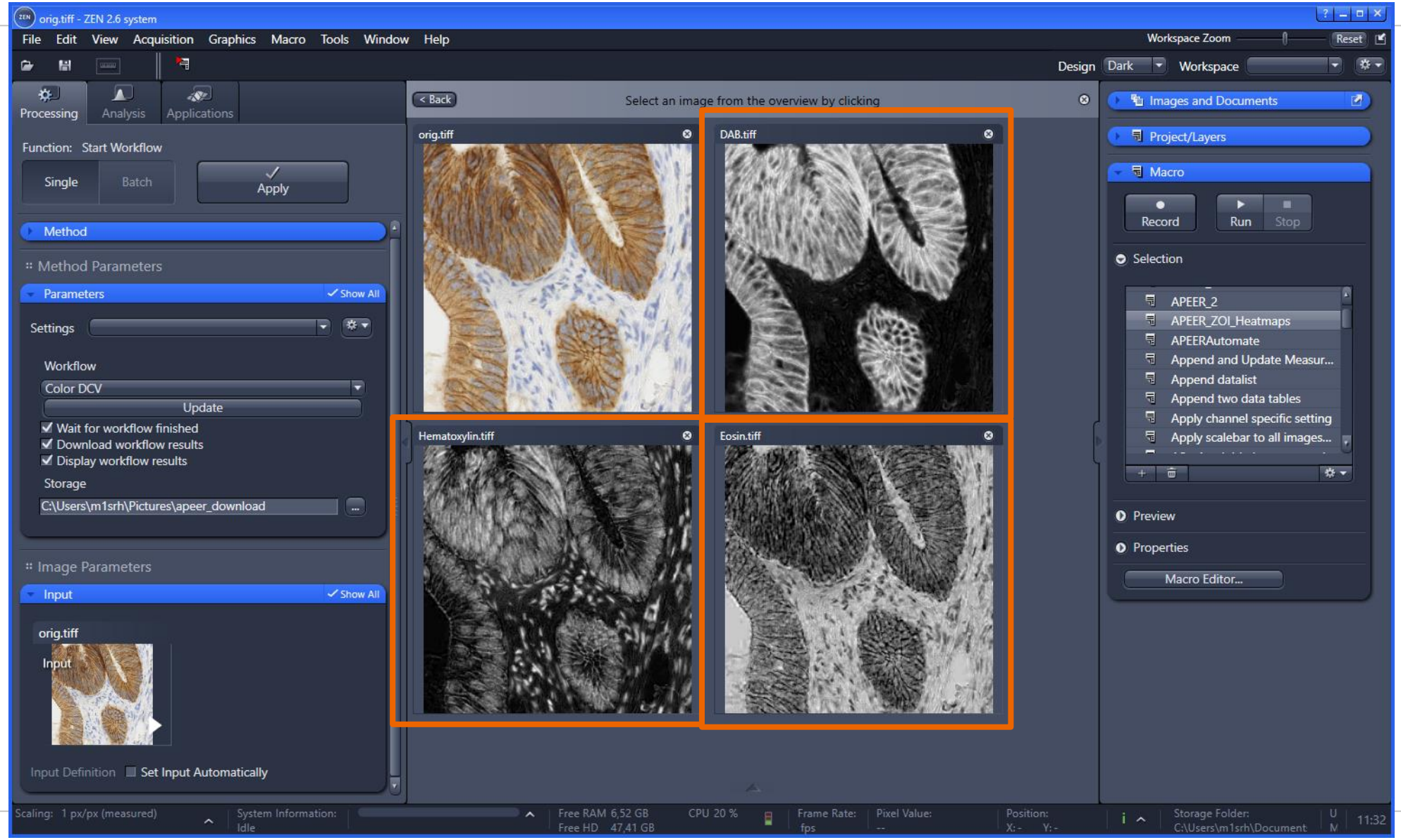
APEER WebViewer - Visualize your images from anywhere using any device



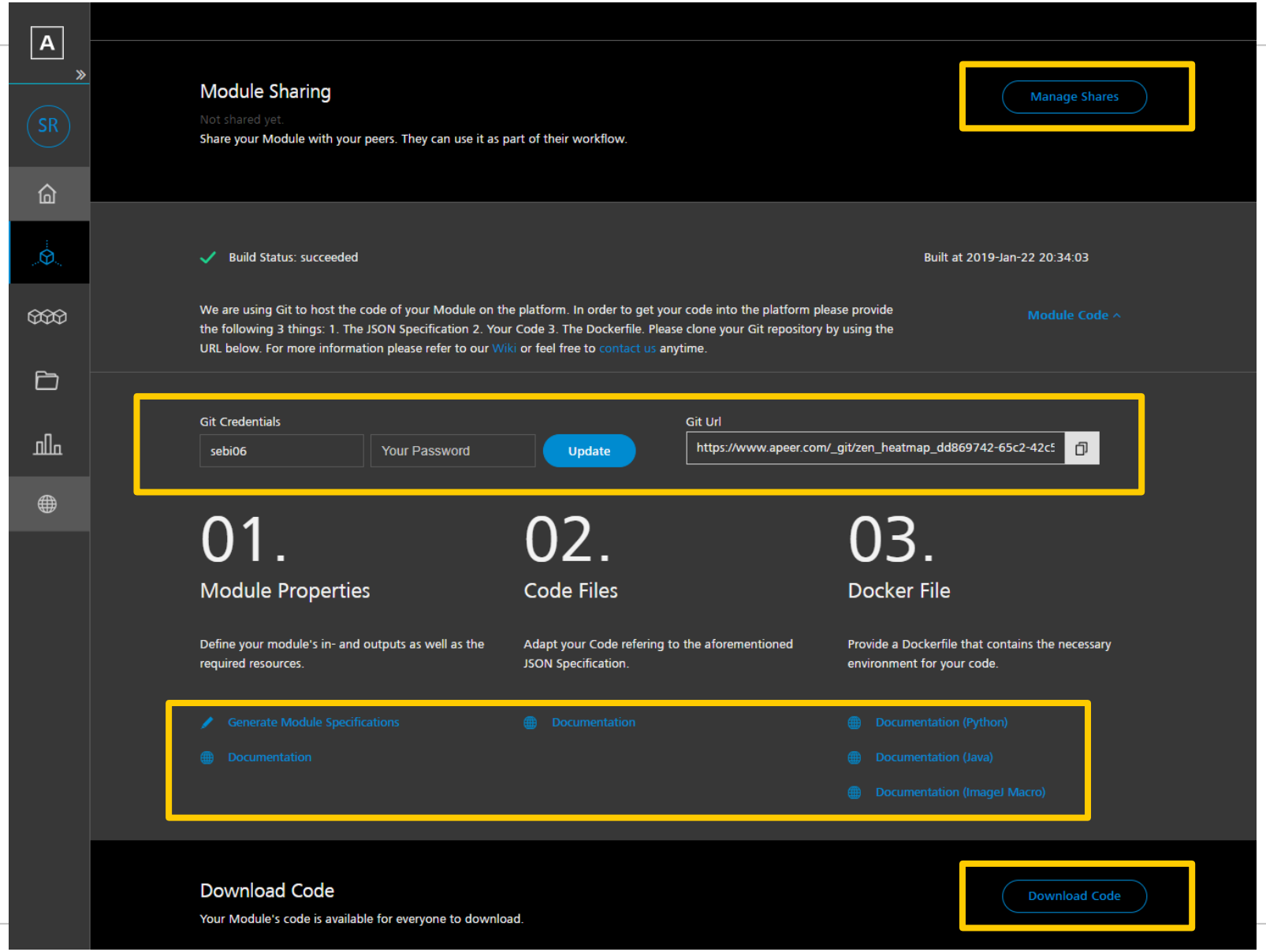
ZEN Blue APEER Connector – Run WorkFlows from “inside” ZEN



ZEN Blue APEER Connector – Run WorkFlows from “inside” ZEN



You have full control over your modules and workflows



- Manage the shares of your modules and workflows
- Use GIT to manage your module code
- Generate modules specs and UI with graphical editor
- Use forum and wiki to find information, tutorials and code examples
- Allow downloading the module code for full transparency

Online Documentation, Tutorials and Videos are available

apeer Docs

Search...

Powered by GitBook

Your First Workflow (10 min)

This step by step tutorial is teaching you how to create a simple workflow running on the APEER platform using a default Python template.

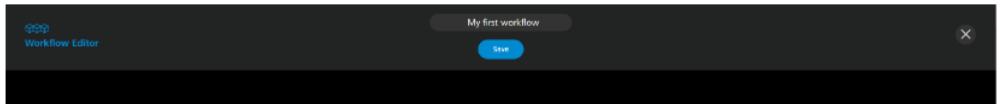
CONTENTS

As we will create a workflow in this tutorial make sure that you've ...

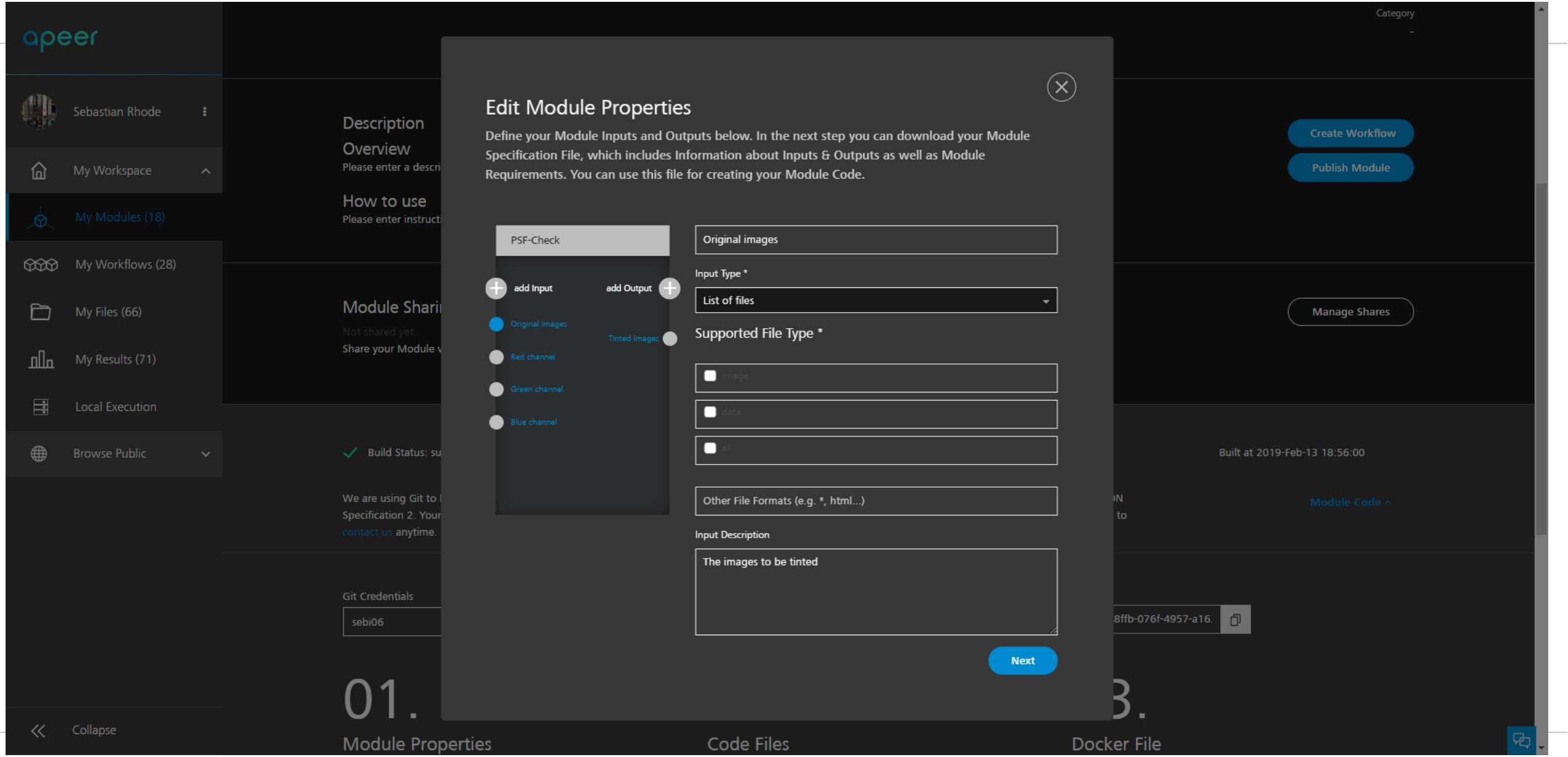
- ... a basic understanding what modules and workflows are
- ... successfully registered an account on APEER
- ... created a first module (use at least the default Python template)

Step #1: Create new workflow with one module

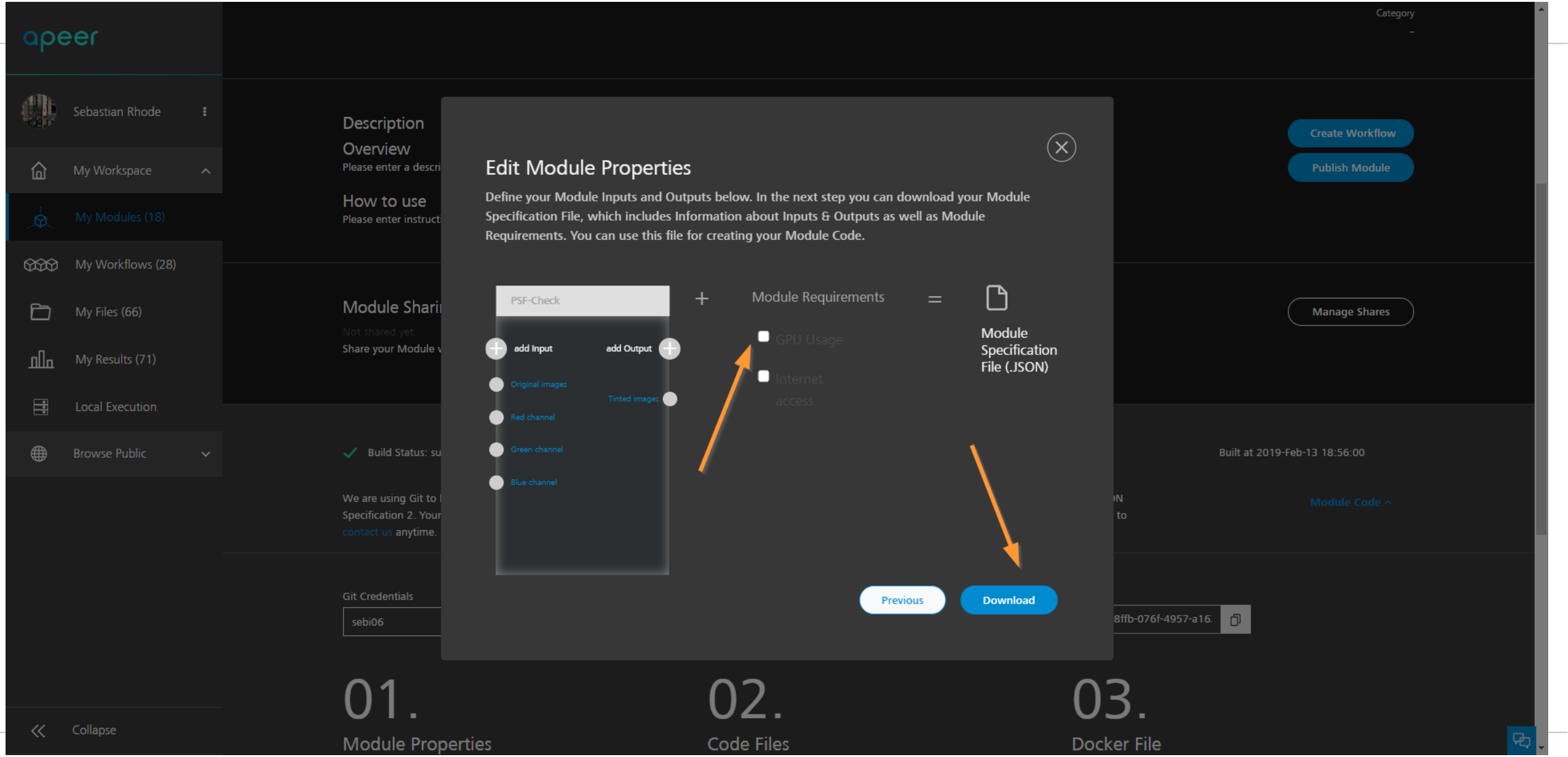
- Go to "My Workspace" and click "Create new Workflow"
- Click the "+" button and type "OME-TIFF-Converter" and click "Add" (This will convert different image files to a standardized OME-TIFF format)
- Click again the "+" button and type "Python"
- Choose the previously created "Python example" module and add it as well
- Enter the name for the workflow e.g. "My first workflow"



Module UI – Use the built-in UI editor



Module UI – Use the built-in UI editor



Module Creation – Code is stored inside a Git Repository

Use IDE of your choice

File Edit Selection View Go Debug Terminal Help — module_specification.json - srh_apeer_sdk_test_362af2b3-aeed-4de3-a857-57a70571a147 - Visual Studio Code

module_specific

Rhode, 21 hours ago | 4 authors (Allan Menoret and others)

```json
{
    "spec": {
        "inputs": {
            "ihc_image": {
                "type:file": {}
            }
        },
        "outputs": {
            "processed_images": {
                "type:list[file]": {}
            }
        }
    },
    "ui": {
        "inputs": {
            "ihc_image": {
                "index": 1,
                "label": "IHC Image",
                "widget:textbox": {}
            }
        },
        "outputs": {}
    }
}
```

Allan Menoret, 7 months ago • Inital

color_dcv.py

You, an hour ago | 2 authors (Rhode and others)

```python
from skimage import color, img_as_float, img_as_uint, io, exposure
from skimage.color import rgb2hed
import numpy as np
import tifffile as tf


def run(original_image_path):

    outputPathList = []

    # Read original image
    ihc_rgb = io.imread(original_image_path)

    # Process
    ihc_hed = rgb2hed(ihc_rgb)
    ihc_h = exposure.rescale_intensity(ihc_hed[:, :, 0], out_range=(0, 255))
    ihc_e = exposure.rescale_intensity(ihc_hed[:, :, 1], out_range=(0, 255))
    ihc_d = exposure.rescale_intensity(ihc_hed[:, :, 2], out_range=(0, 255))

    images = [ihc_h, ihc_e, ihc_d]
    outnames = ['Hematoxylin.tiff', 'Eosin.tiff', 'DAB.tiff']

    for img, out in zip(images, outnames):

        outputPathList.append(out)
        tf.imwrite(out, np.int16(img))

    tf.imwrite('HED.tiff', np.float16(ihc_hed))
    outputPathList.append('HED.tiff')

    print(outputPathList)

    # Return apeer output values as dictionary
    return {'processed_images': outputPathList}
```

apeer_main.py

You, 6 hours ago | 3 authors (Rhode and others)

```python
from apeer_dev_kit import adk
import color_dcv

if __name__ == "__main__":

    inputs = adk.get_inputs()
    outputs = color_dcv.run(inputs['ihc_image'])
    adk.set_file_output('processed_images', outputs['processed_images'])
    adk.finalize()

```

requirements.txt

You, an hour ago | 2 authors (Thomas Irmer and others)

```
apeer-dev-kit>=1,<2
cloudpickle==0.6.1
dask==0.20.2
decorator==4.3.0
networkx==2.2
numpy==1.15.4
Pillow==5.3.0
PyWavelets==1.0.1
scikit-image==0.14.1
six==1.11.0
toolz==0.9.0
scipy==1.1.0
tifffile==2018.11.28
```

master* Python 3.7.0 64-bit ('Anaconda3': conda) Team 1 2 Allan Menoret, 7 months ago Ln 24, Col 2 Spaces: 4 UTF-8 CRLF JSON

Module Creation - GIT

Specify module UI and other requirements (in this case Python modules)

File Edit Selection View Go Debug module_specification.json - srh_apeer_sdk_test_362af2b3-aeed-4de3-a857-57a70571a147 - V...

module_specification.json

Rhode, 21 hours ago | 4 authors (Allan Menoret and others)

```
{
    "spec": {
        "inputs": {
            "ihc_image": {
                "type:file": {}
            }
        },
        "outputs": {
            "processed_images": {
                "type:list[file]": {}
            }
        }
    },
    "ui": {
        "inputs": {
            "ihc_image": {
                "index": 1,
                "label": "IHC Image",
                "widget:textbox": {}
            }
        },
        "outputs": {}
    }
}
```

Allan Menoret, 7 months ago • Inital commit tint mo

requirements.txt

You, an hour ago | 2 authors (Thomas Irmer and others)

```
apeer-dev-kit>=1,<2
cloudpickle==0.6.1
dask==0.20.2
decorator==4.3.0
networkx==2.2
numpy==1.15.4
Pillow==5.3.0
PyWavelets==1.0.1
scikit-image==0.14.1
six==1.11.0
toolz==0.9.0
scipy==1.1.0
tifffile==2018.11.28
```

master* Team Python 3.7.0 64-bit ('Anaconda3': conda) 0 0 Ln 24, Col 2 Spaces: 4 UTF-8 CRLF JSON

Module Creation – APEER-SDK

Specify module UI and other requirements (in this case Python modules)

Search projects

Help Donate Log in Register

apeer-dev-kit 1.0.9

```
pip install apeer-dev-kit
```

Latest version

Last released: Dec 21, 2018

Development kit for creating modules on apeer

Navigation

- Project description
- Release history
- Download files

Project links

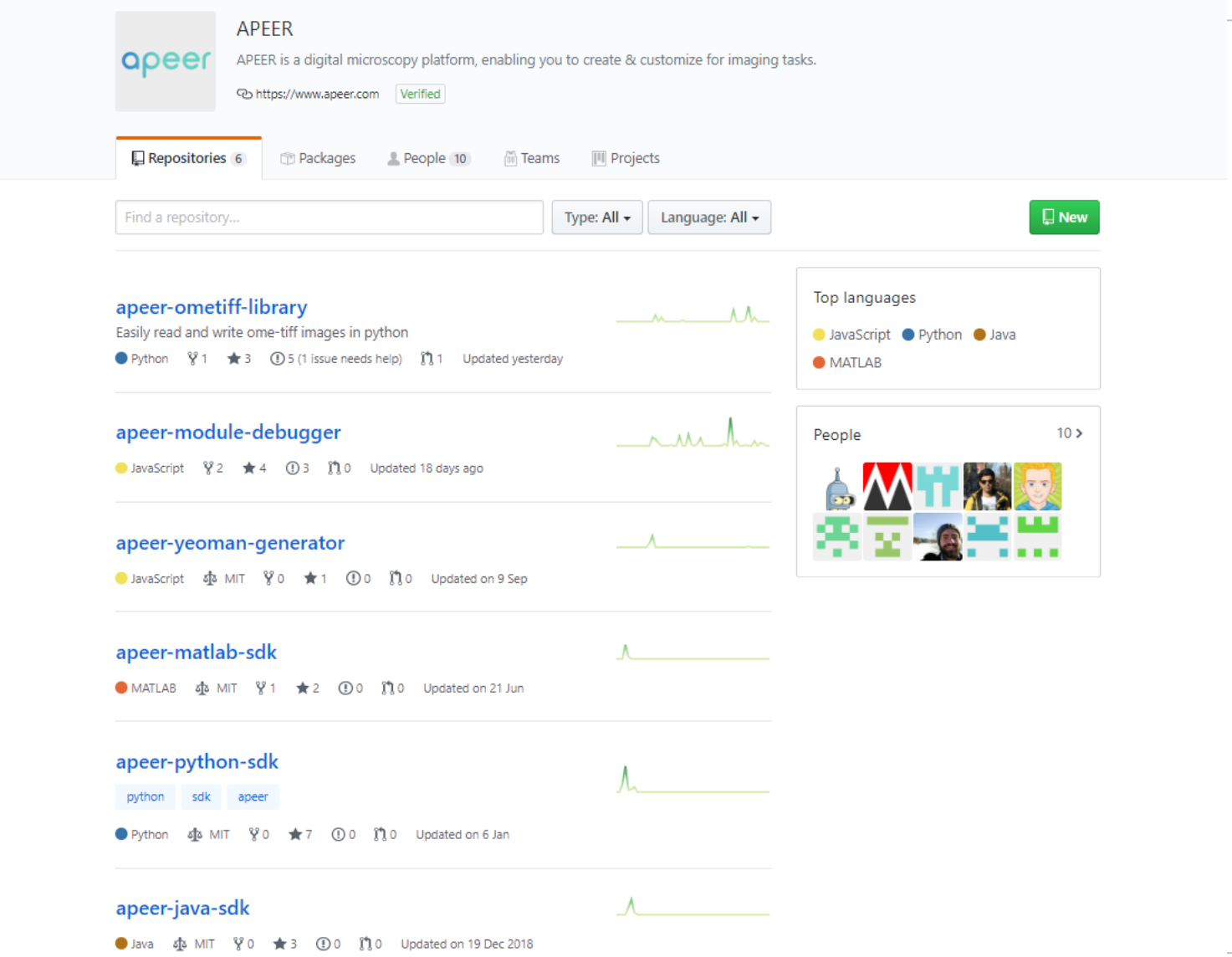
- Homepage

Project description

APEER Python SDK aka (ADK) is a Python library for reading inputs and writing outputs of APEER(https://www.apeer.com) modules. The ADK will take care of reading inputs from previous modules in APEER and writing your outputs in the correct format for the next mod ule. This project is hosted at https://github.com/apeer-micro/apeer-python-sdk The documentation can be found at https://github.com/apeer-micro/apeer-python-sdk/blob/master/README.md

Module Creation – APEER-SDK on GitHub

Specify module UI and other requirements (in this case Python modules)



Module Creation - GIT

Write your code and use the APEER-SDK (PyPi) to make life easy

File Edit Selection View Go Debug Terminal Help apeer_main.py - srh_apeer_sdk_test_362af2b3-aeed-4de3-a857-57a70571a147 - Visual Studio Code

color_dcv.py

```python
from skimage import color, img_as_float, img_as_uint, io, exposure
from skimage.color import rgb2hed
import numpy as np
import tifffile as tf


def run(original_image_path):

    outputPathList = []

    # Read original image
    ihc_rgb = io.imread(original_image_path)

    # Process
    ihc_hed = rgb2hed(ihc_rgb)
    ihc_h = exposure.rescale_intensity(ihc_hed[:, :, 0], out_range=(0, 255))
    ihc_e = exposure.rescale_intensity(ihc_hed[:, :, 1], out_range=(0, 255))
    ihc_d = exposure.rescale_intensity(ihc_hed[:, :, 2], out_range=(0, 255))

    images = [ihc_h, ihc_e, ihc_d]
    outnames = ['Hematoxylin.tiff', 'Eosin.tiff', 'DAB.tiff']

    for img, out in zip(images, outnames):

        outputPathList.append(out)
        tf.imwrite(out, np.int16(img))

    tf.imwrite('HED.tiff', np.float16(ihc_hed))
    outputPathList.append('HED.tiff')

    print(outputPathList)

    # Return apeer output values as dictionary
    return {'processed_images': outputPathList}

```

apeer_main.py

```python
from apeer_dev_kit import adk
import color_dcv

if __name__ == "__main__":

    inputs = adk.get_inputs()
    outputs = color_dcv.run(inputs['ihc_image'])
    adk.set_file_output('processed_images', outputs['processed_images'])
    adk.finalize()

```

master* Team Python 3.7.0 64-bit ('Anaconda3': conda) 1 1 Ln 10, Col 1 Spaces: 4 UTF-8 with BOM CRLF Python

Buisness Model

Pricing

There will always be a free version - APEER Core. Due to our users' higher demand in storage and processing power, we are working on a subscription based Pro version, which will be rolled out soon.

Please do not hesitate to contact us at support@apeer.com

	APEER CORE Best package to get started for individuals	APEER PRO Fast processing and high storage	APEER ACADEMIC Hybrid solution with local processing and local data or individual cloud storage	APEER CUSTOM Get customized solutions for your image processing problem (industry and academia).
Number of users	1	1	10	customizable
Online storage	100 GB	1 TB	customizable	customizable
Local storage			✓	✓
Normal processing	✓			
Fast processing		✓	✓	✓
Queue priority		✓	✓	✓
E-Mail support	✓	✓	✓	✓
Community support	✓	✓	✓	✓
Phone support		✓	✓	✓
Personal advise support				✓
Build Modules / Workflows	✓	✓	✓	✓
Share with individuals	✓	✓	✓	✓
Share with subgroups			✓	✓
Get a customized Workflow				✓
	FREE	Coming soon	Coming soon	

ZEISS