



Getting started with NaaVRE

G. Pelouze, S. Koulouzis, Z. Zhao

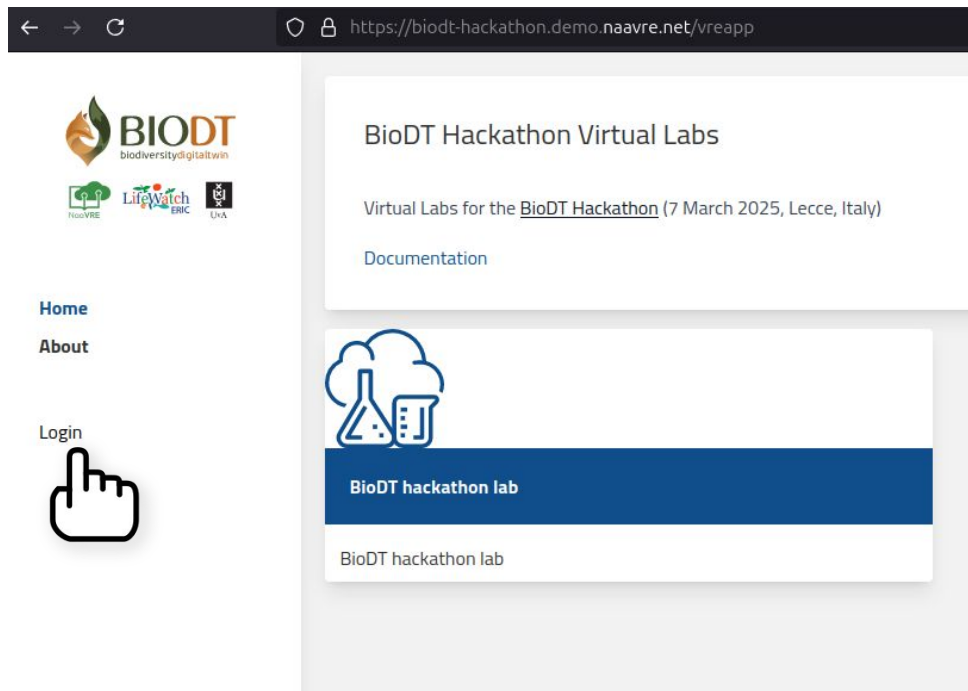
Starting point

Get started at

<https://biodyt-hackathon.demo.naavre.net/vreapp>



Sign in



← → ↻ <https://biodyt-hackathon.demo.naavre.net/vreapp>

BIODT
biodiversitydigitaltwin

[Home](#)
[About](#)

Login 

BioDT Hackathon Virtual Labs

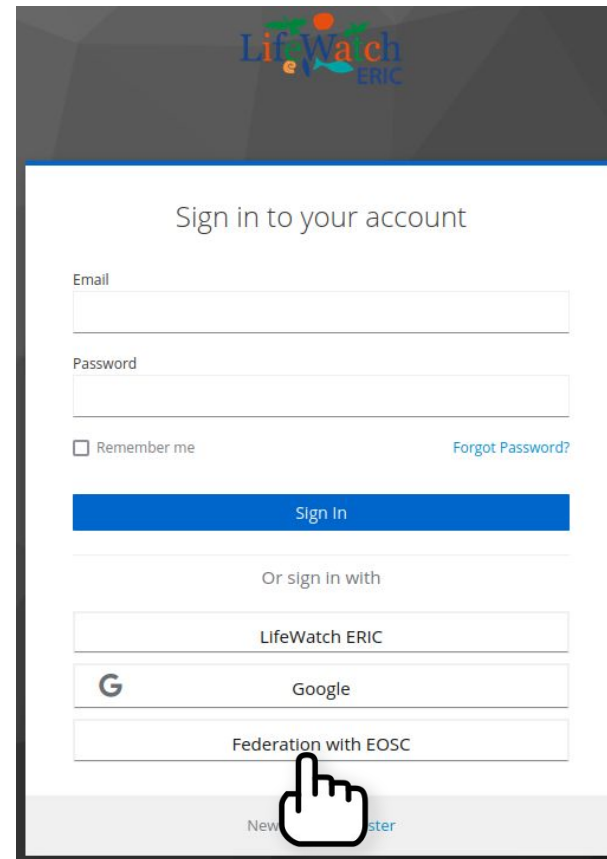
Virtual Labs for the [BioDT Hackathon](#) (7 March 2025, Lecce, Italy)

[Documentation](#)



BioDT hackathon lab

BioDT hackathon lab





Sign in to your account

Email

Password

☐ Remember me [Forgot Password?](#)

Sign In

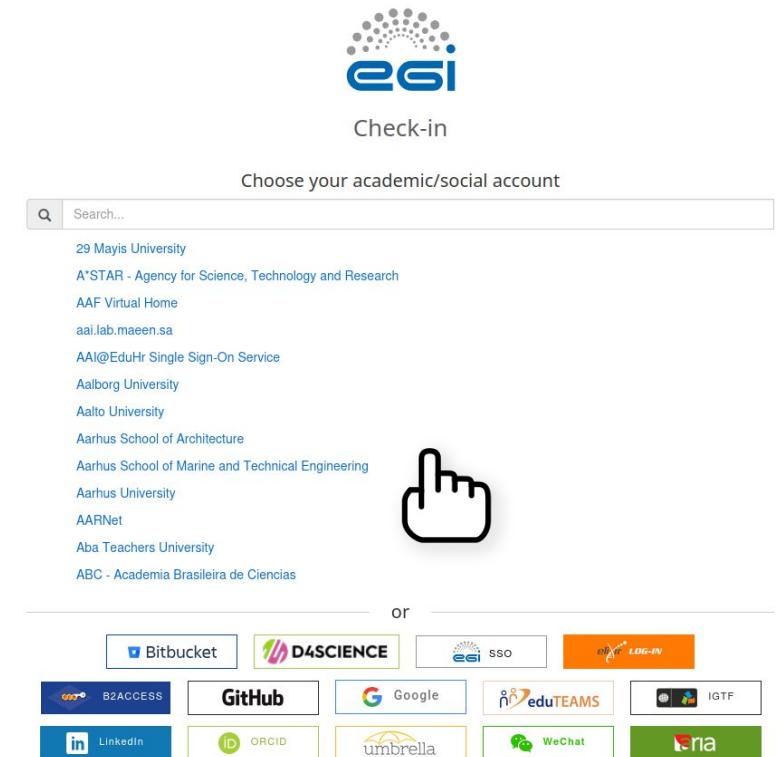
Or sign in with

LifeWatch ERIC

 Google

Federation with EOSC

New  [Register](#)





Check-in

Choose your academic/social account

Search...

- [29 Mayis University](#)
- [A*STAR - Agency for Science, Technology and Research](#)
- [AAF Virtual Home](#)
- [aai.lab.maeen.sa](#)
- [AAI@EduHr Single Sign-On Service](#)
- [Aalborg University](#)
- [Aalto University](#)
- [Aarhus School of Architecture](#)
- [Aarhus School of Marine and Technical Engineering](#)
- [Aarhus University](#)
- [AARNet](#)
- [Aba Teachers University](#)
- [ABC - Academia Brasileira de Ciencias](#)

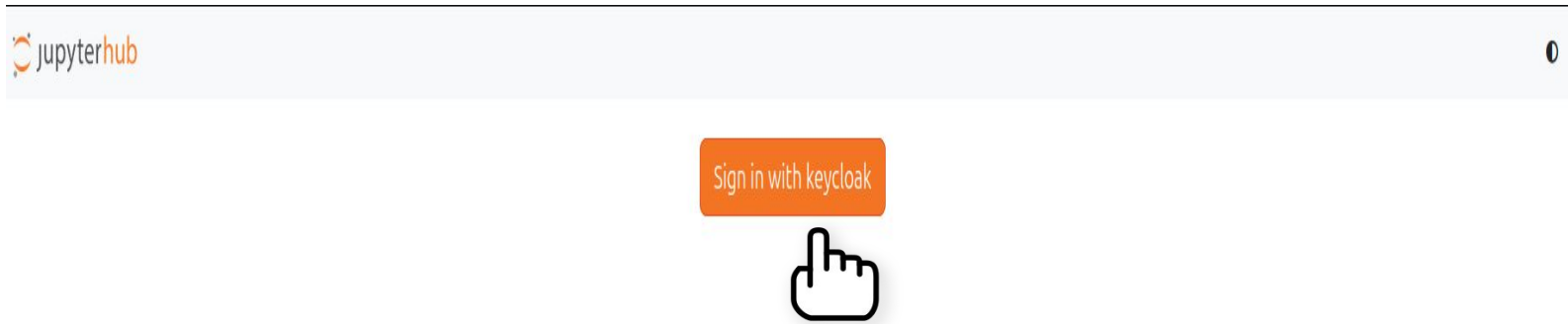
or

Start your virtual lab



The NaaVRE interface

Manage local files

Explore public
GitHub repositories

Manage your
work with Git

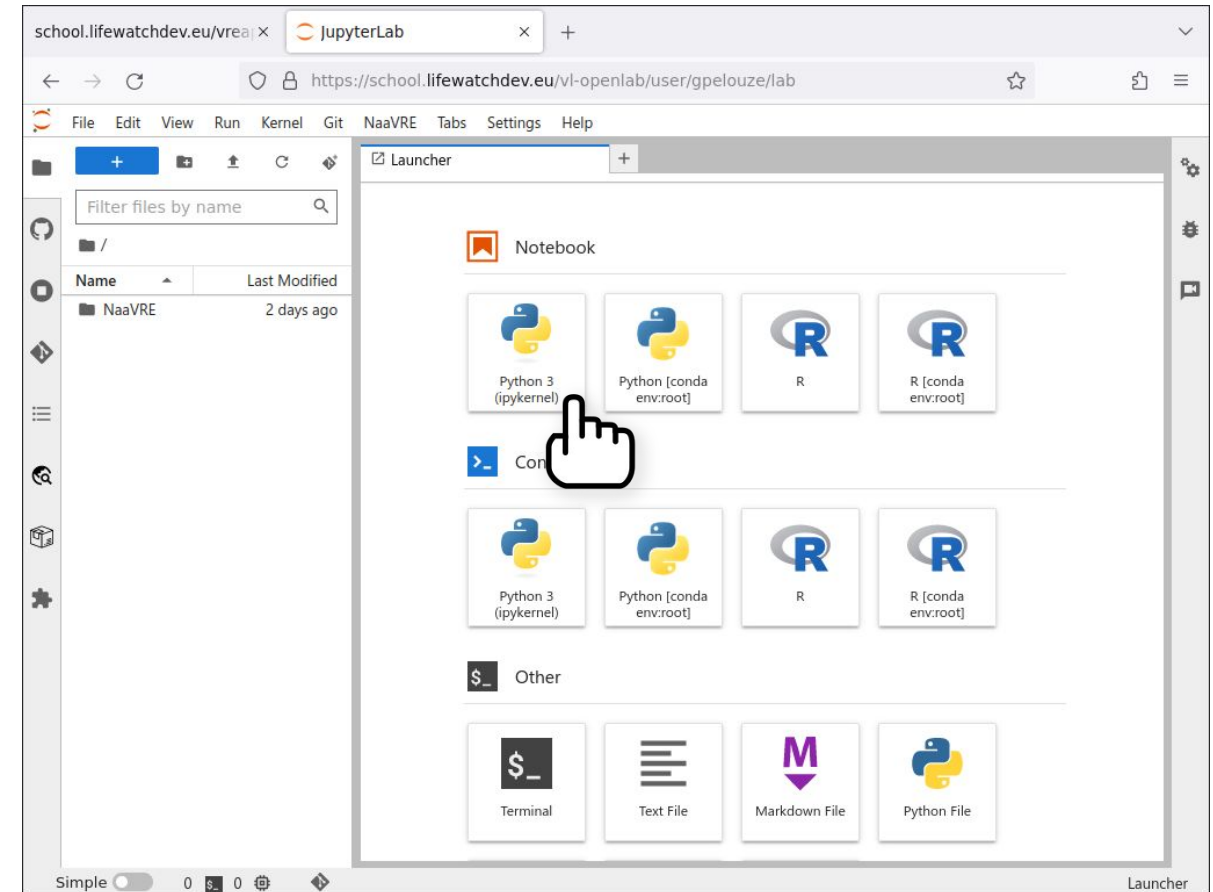
Find notebooks

Containerize cells

The screenshot displays the NaaVRE JupyterLab interface within a web browser. The browser's address bar shows the URL: <https://school.lifewatchdev.eu/vl-openlab/user/gpelouze/lab/workspaces/auto-a/tree/NaaVRE>. The interface includes a top menu bar with options: File, Edit, View, Run, Kernel, Git, NaaVRE, Tabs, Settings, and Help. A left sidebar contains icons for file management, GitHub exploration, Git management, notebook search, and cell containerization. The main workspace is titled 'NaaVRE' and features a 'Launcher' tab. It is organized into sections: 'Notebook' (with Python 3 (ipykernel), Python (conda env:root), R, and R (conda env:root) options), 'Console' (with similar Python and R options), 'Other' (including Terminal, Text File, Markdown File, Python File, R File, Show Contextual Help, and Video Chat), and 'VRE Components' (featuring the 'Experiment Manager' icon). A teal callout box 'Create Python or R notebooks' points to the notebook options, and another teal callout box 'Create and run workflows' points to the 'Experiment Manager' icon. The bottom status bar shows 'Simple' mode, a '0' count, and a 'Launcher' label.

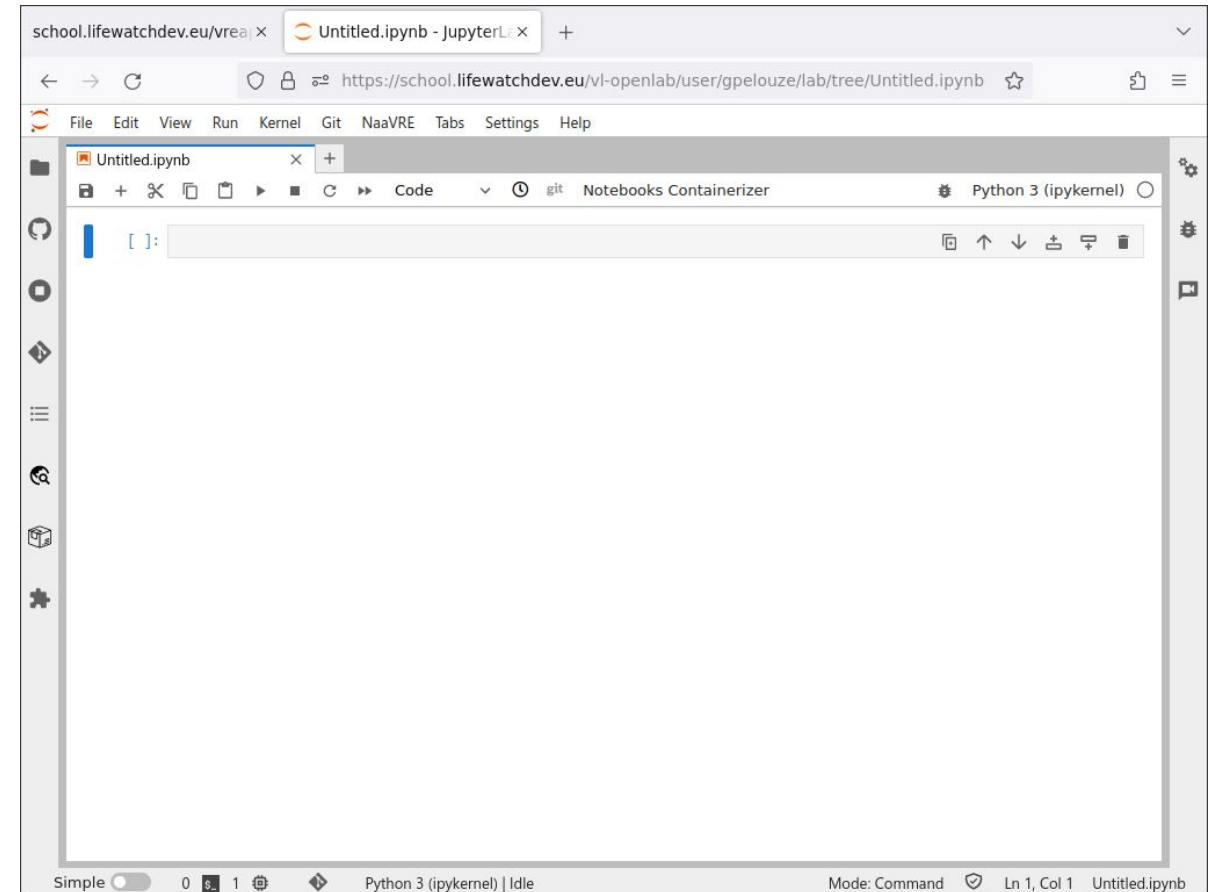
Create a notebook

1. Create a new notebook



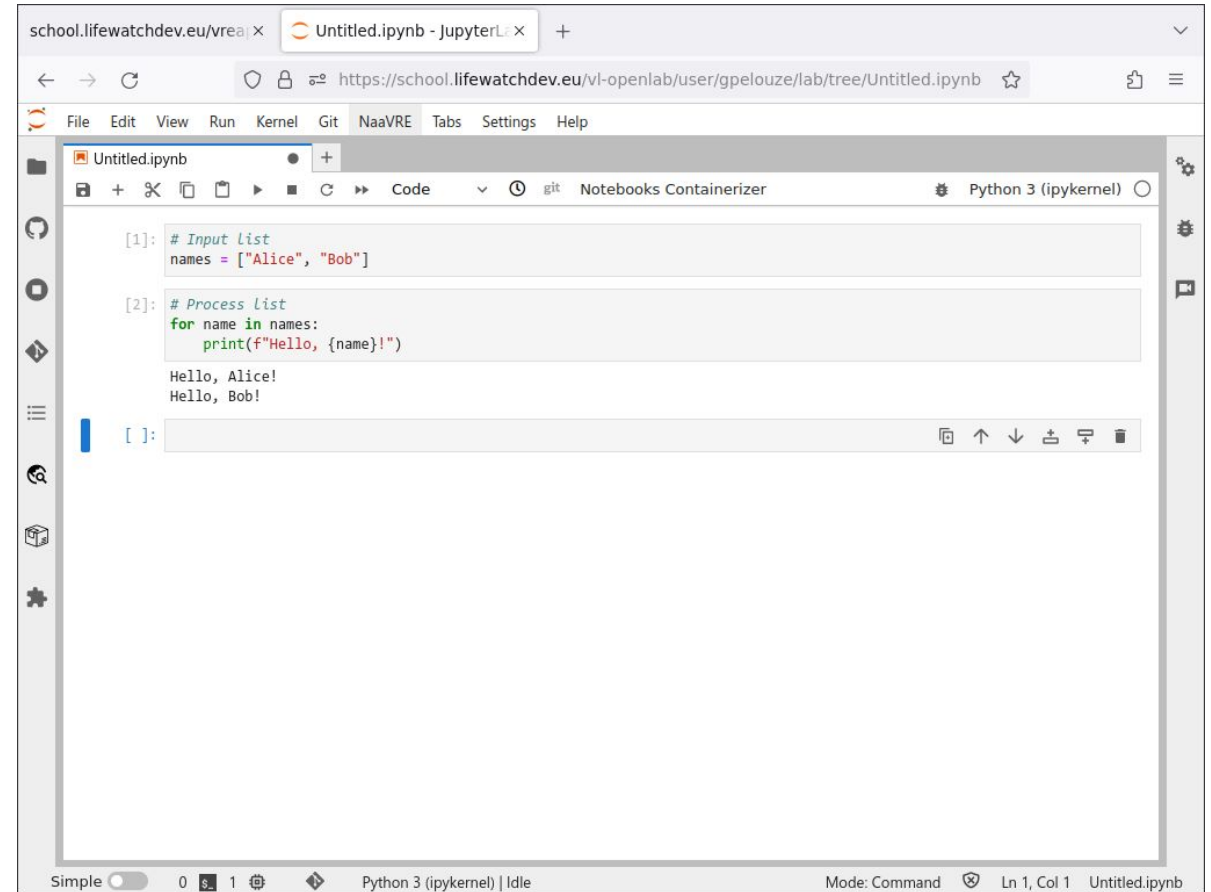
Create a notebook

1. Create a new notebook



Create a notebook

1. Create a new notebook
2. Write some code



The screenshot shows a JupyterLab notebook titled 'Untitled.ipynb' in a web browser. The notebook contains two code cells. The first cell, labeled '[1]:', contains the following Python code:

```
# Input List
names = ["Alice", "Bob"]
```

The second cell, labeled '[2]:', contains the following Python code:

```
# Process List
for name in names:
    print(f"Hello, {name}!")
```

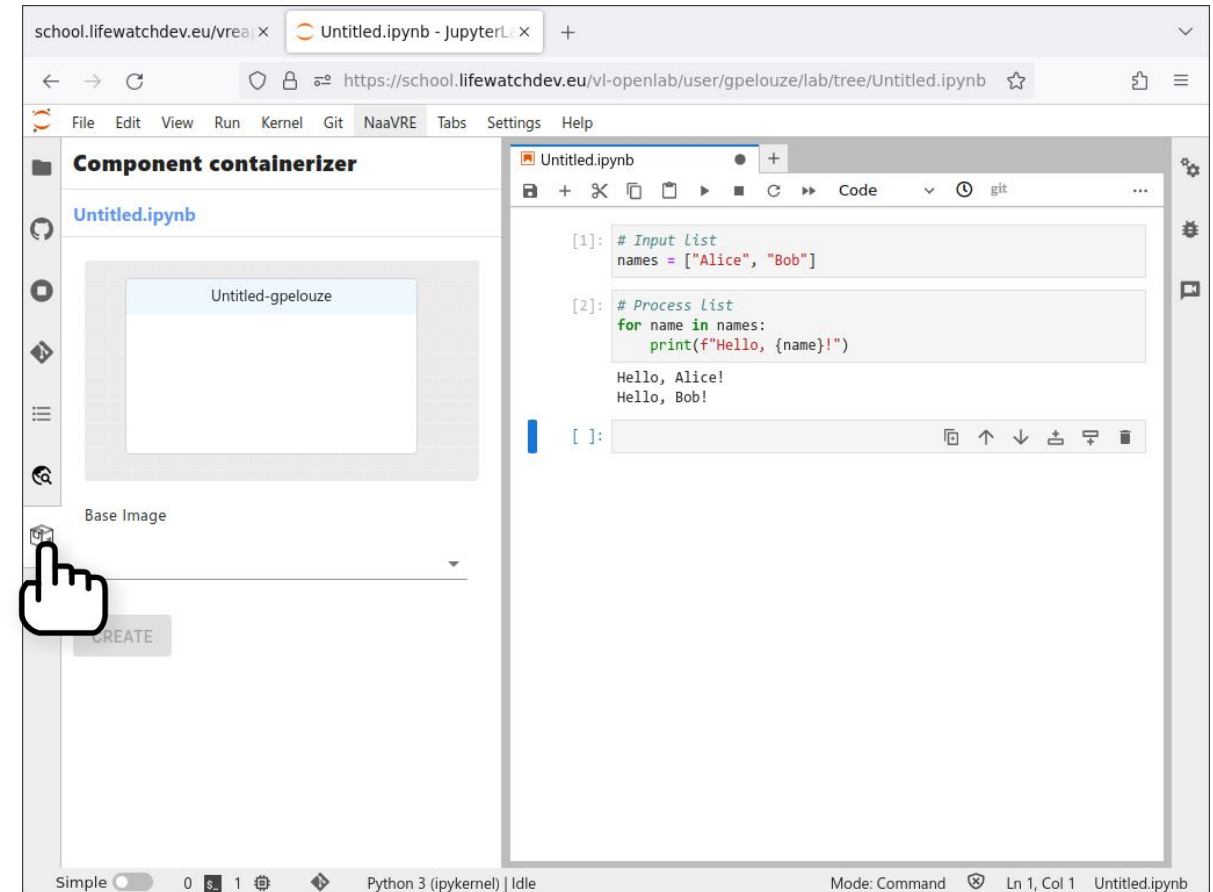
The output of the second cell is displayed below the code:

```
Hello, Alice!
Hello, Bob!
```

The notebook interface includes a menu bar (File, Edit, View, Run, Kernel, Git, NaaVRE, Tabs, Settings, Help) and a toolbar with various icons for file operations and execution. The status bar at the bottom indicates the notebook is running on Python 3 (ipykernel) in 'Simple' mode, with the cursor at line 1, column 1.

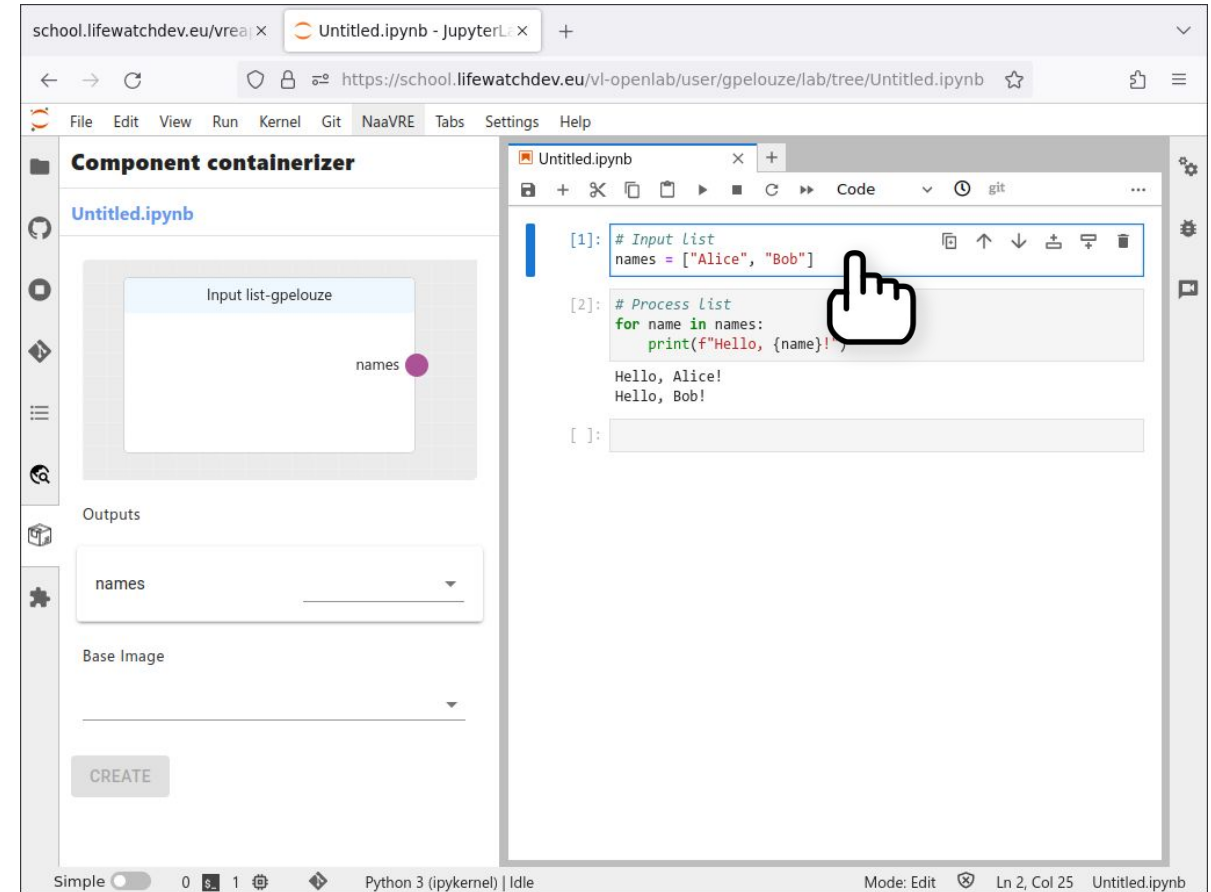
Containerize the cells

1. Create a new notebook
2. Write some code
3. Containerize the cells
 - a. open the component containerizer



Containerize the cells

1. Create a new notebook
2. Write some code
3. Containerize the cells
 - a. open the component containerizer
 - b. select the cell you want to containerize



The screenshot shows the JupyterLab interface with the 'Component containerizer' panel on the left and a code cell being selected in the main editor. The code cell contains the following code:

```
[1]: # Input List
names = ["Alice", "Bob"]

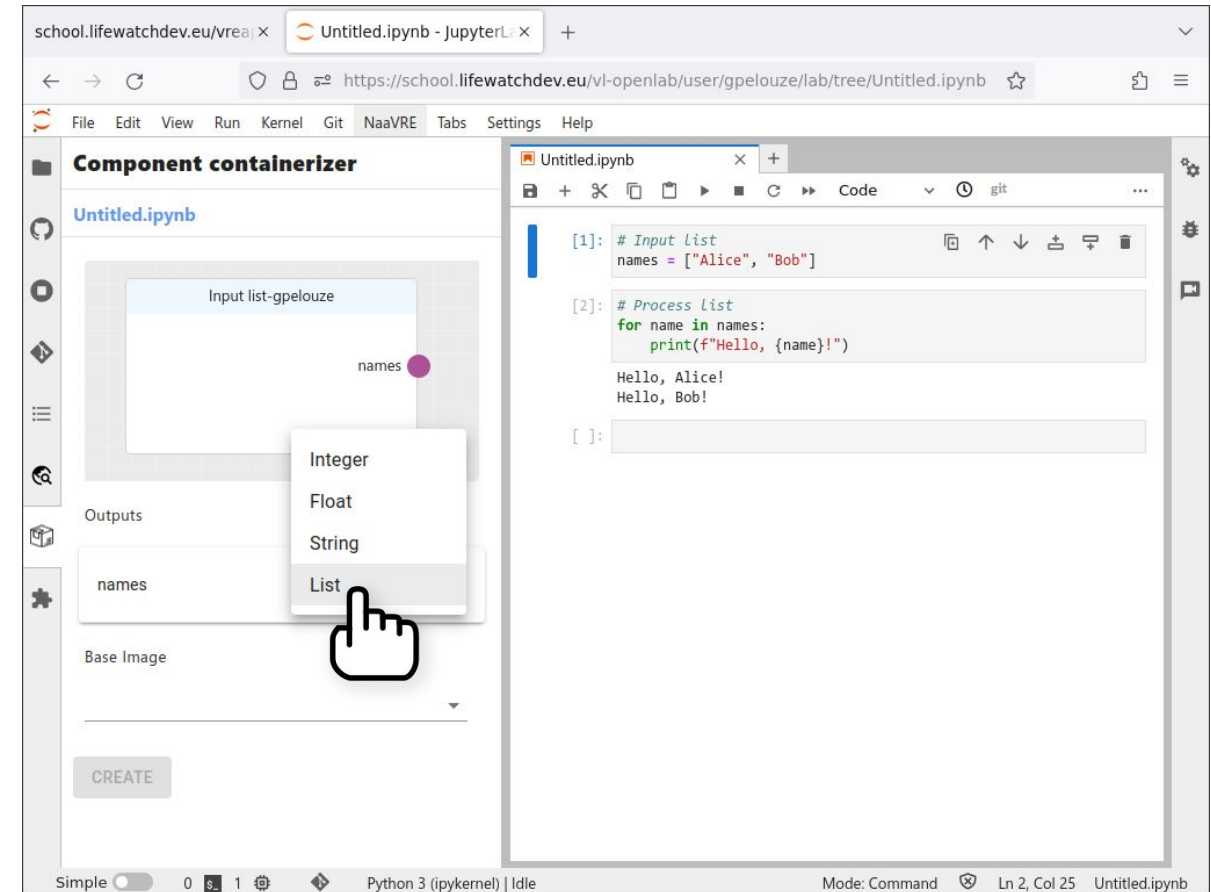
[2]: # Process List
for name in names:
    print(f"Hello, {name}!")

Hello, Alice!
Hello, Bob!
```

A hand cursor is pointing at the first code cell, indicating it is being selected for containerization. The 'Component containerizer' panel on the left shows a diagram of the component containerizer with a box labeled 'Input list-gpelouze' and a variable 'names' connected to it. Below the diagram, there is a dropdown menu for 'names' and a 'CREATE' button.

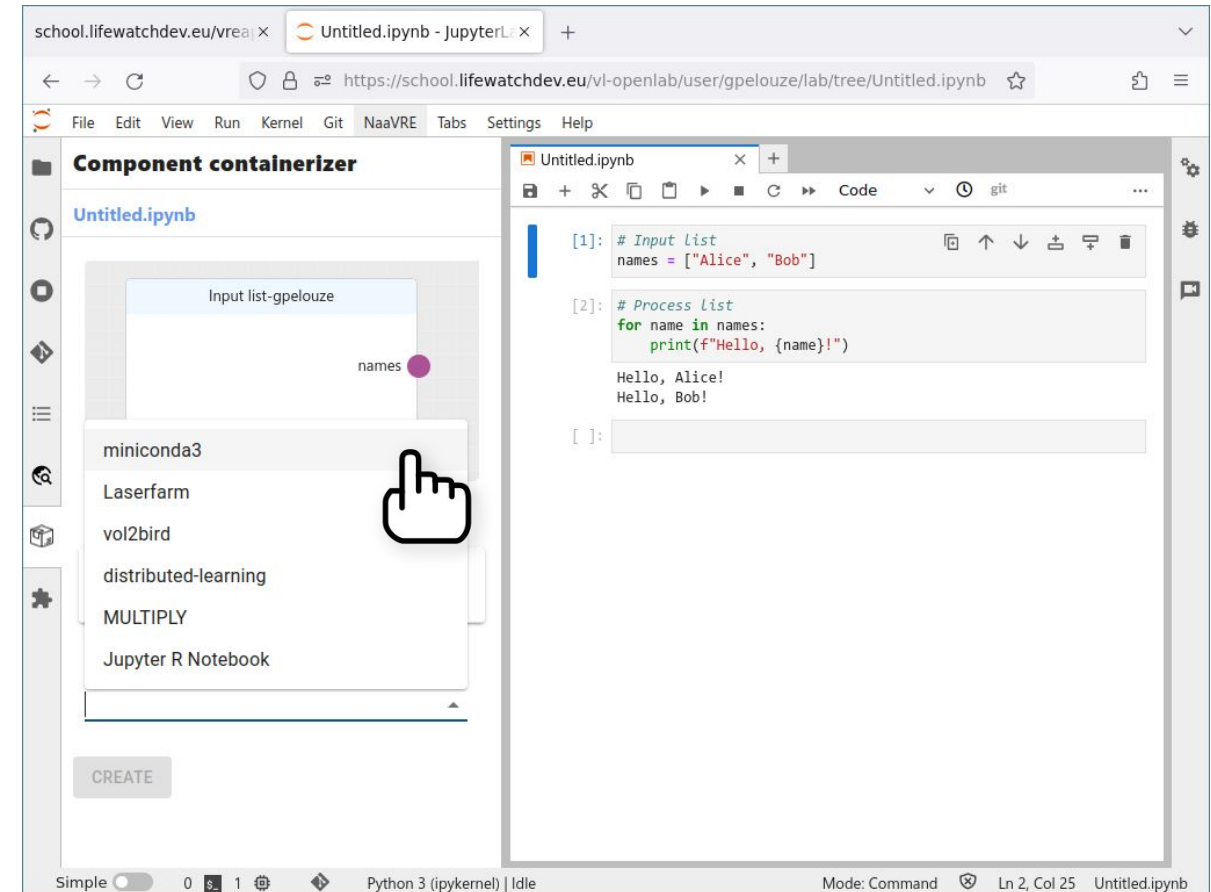
Containerize the cells

1. Create a new notebook
2. Write some code
3. Containerize the cells
 - a. open the component containerizer
 - b. select the cell you want to containerize
 - c. set the variable types



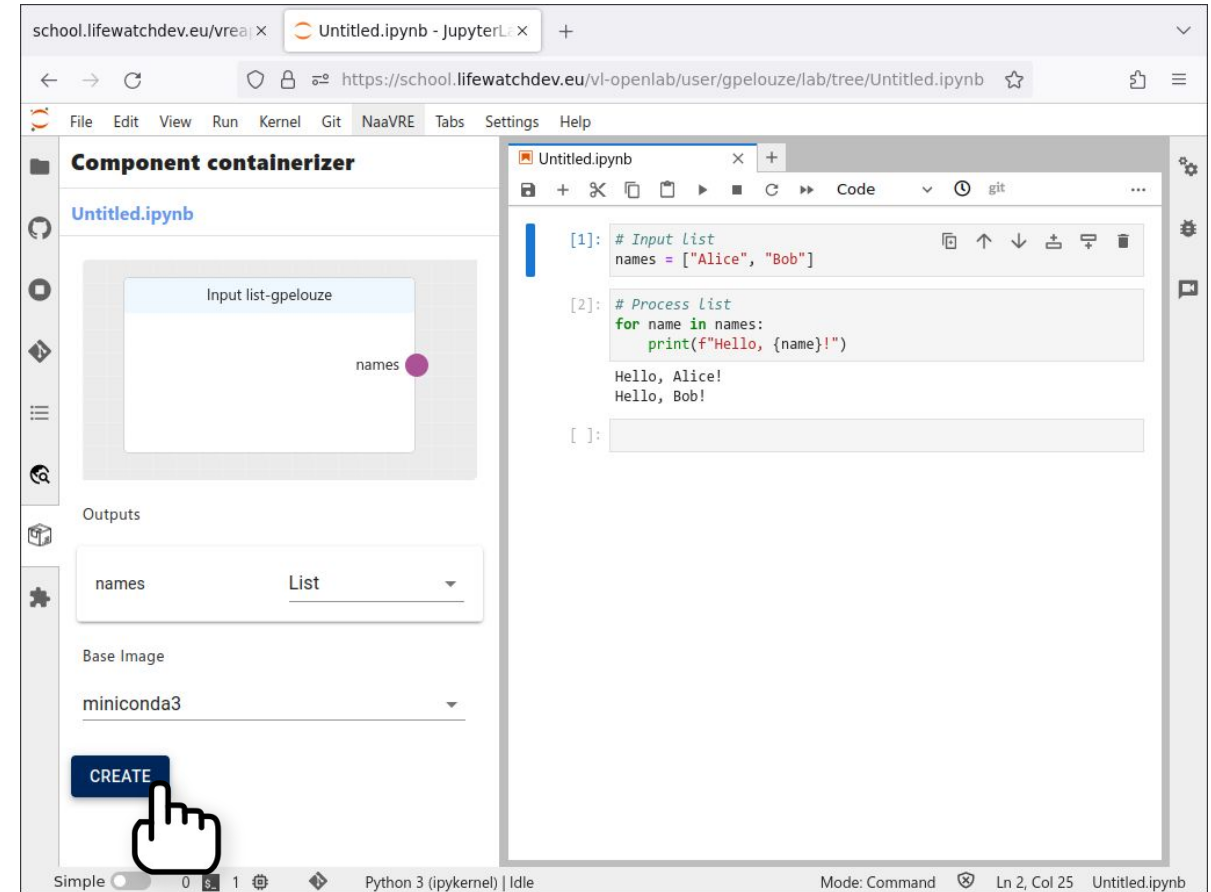
Containerize the cells

1. Create a new notebook
2. Write some code
3. Containerize the cells
 - a. open the component containerizer
 - b. select the cell you want to containerize
 - c. set the variable types
 - d. select the base image



Containerize the cells

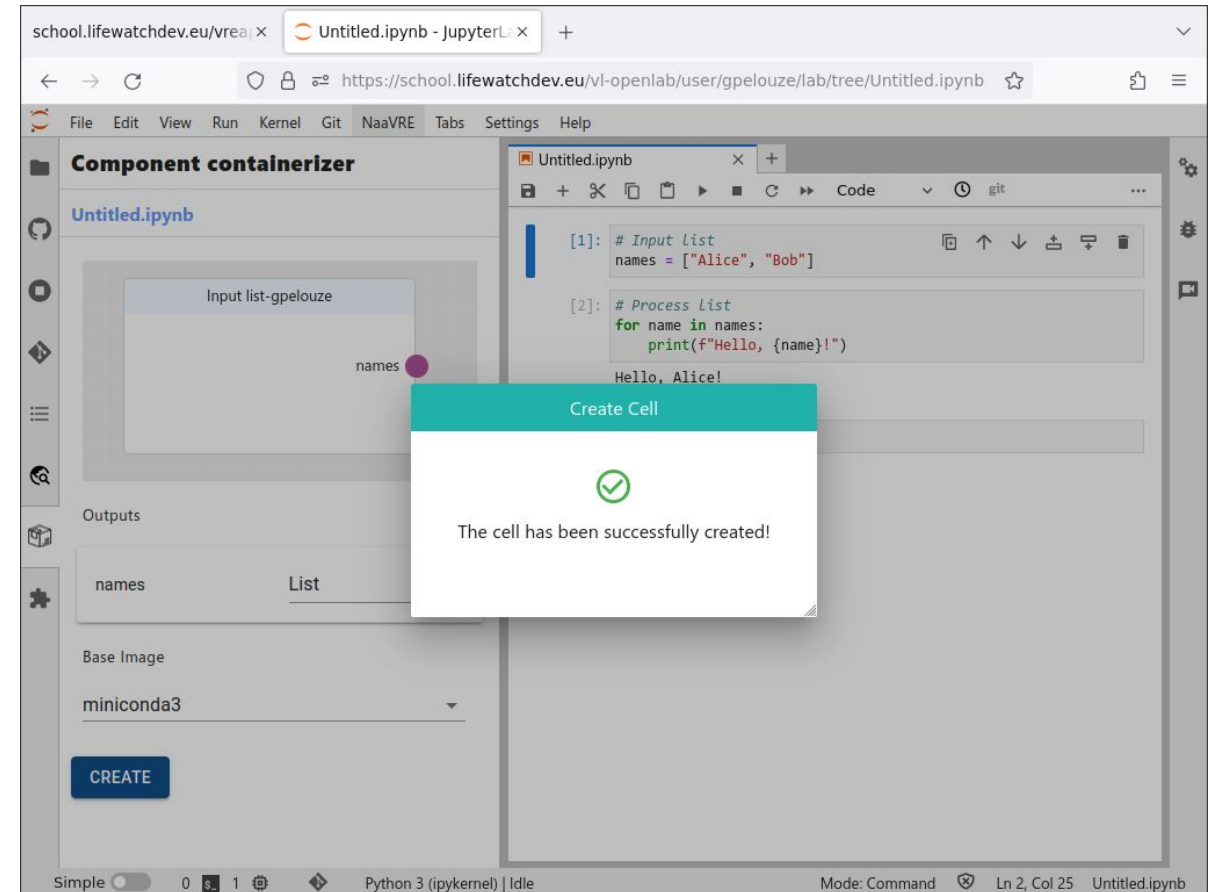
1. Create a new notebook
2. Write some code
3. Containerize the cells
 - a. open the component containerizer
 - b. select the cell you want to containerize
 - c. set the variable types
 - d. select the base image
 - e. create the cell



The screenshot shows the JupyterLab interface with the 'Component containerizer' panel on the left and a code editor on the right. The containerizer panel displays a diagram of a component with an input list and a process list. Below the diagram, there are dropdowns for 'names' (set to 'List') and 'Base Image' (set to 'miniconda3'). A 'CREATE' button is at the bottom of the panel, with a hand cursor pointing to it. The code editor on the right contains two code cells. The first cell defines a list 'names' with 'Alice' and 'Bob'. The second cell is a process list that iterates over 'names' and prints 'Hello, {name}!' for each name. The output of the second cell shows 'Hello, Alice!' and 'Hello, Bob!'.

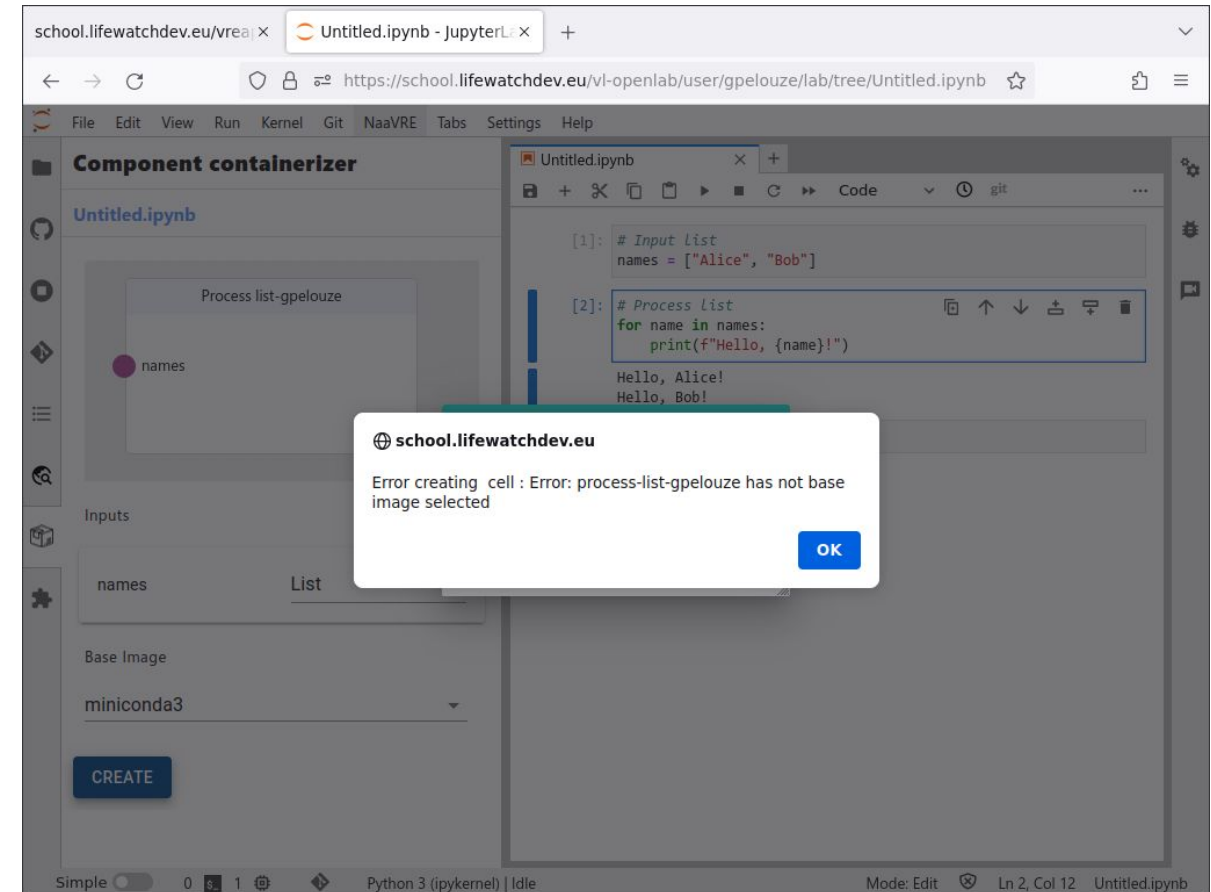
Containerize the cells

1. Create a new notebook
2. Write some code
3. Containerize the cells
 - a. open the component containerizer
 - b. select the cell you want to containerize
 - c. set the variable types
 - d. select the base image
 - e. create the cell
 - f. well done!
 - g. repeat steps b–f for other cells



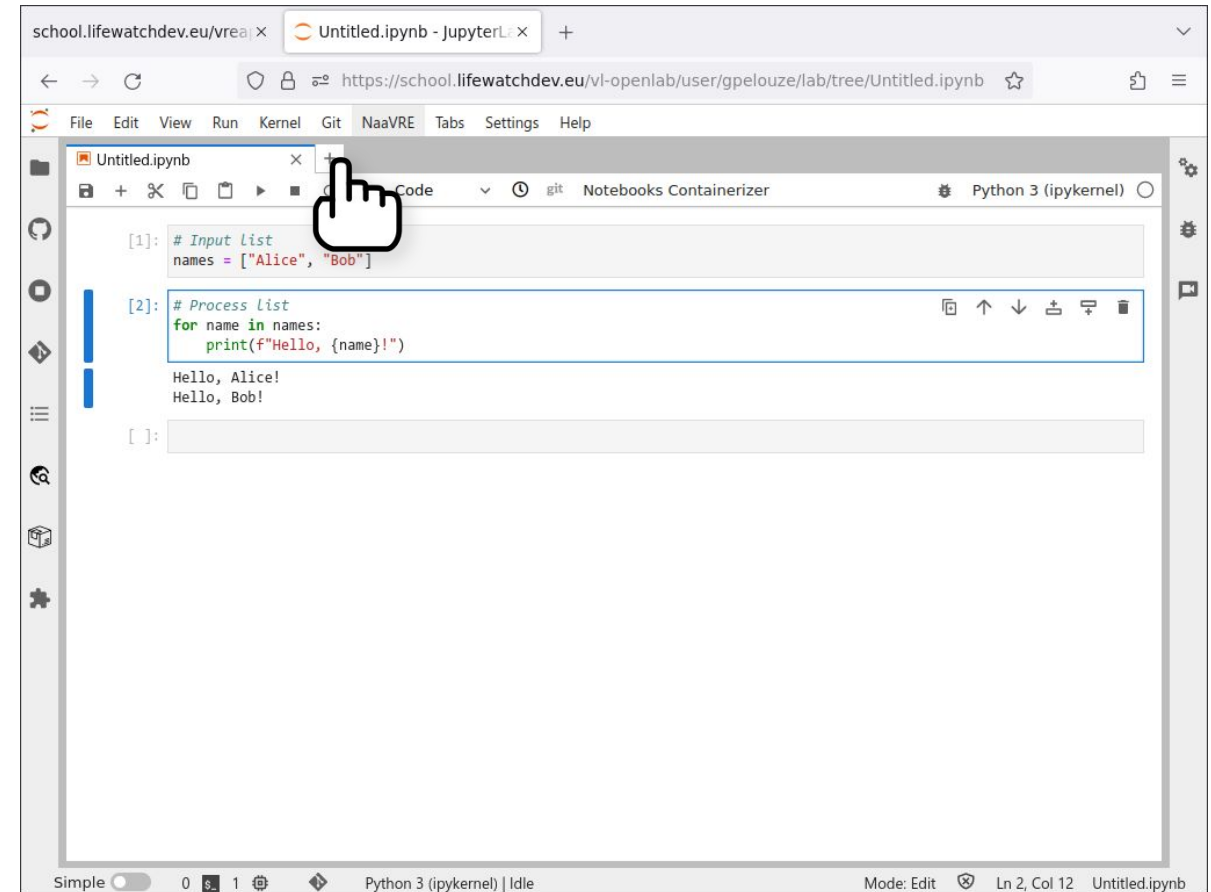
Containerize the cells (troubleshooting)

If you get the message “Error creating cell: no base image selected”, reset the base image by clicking on ✕ and selecting it again from the list



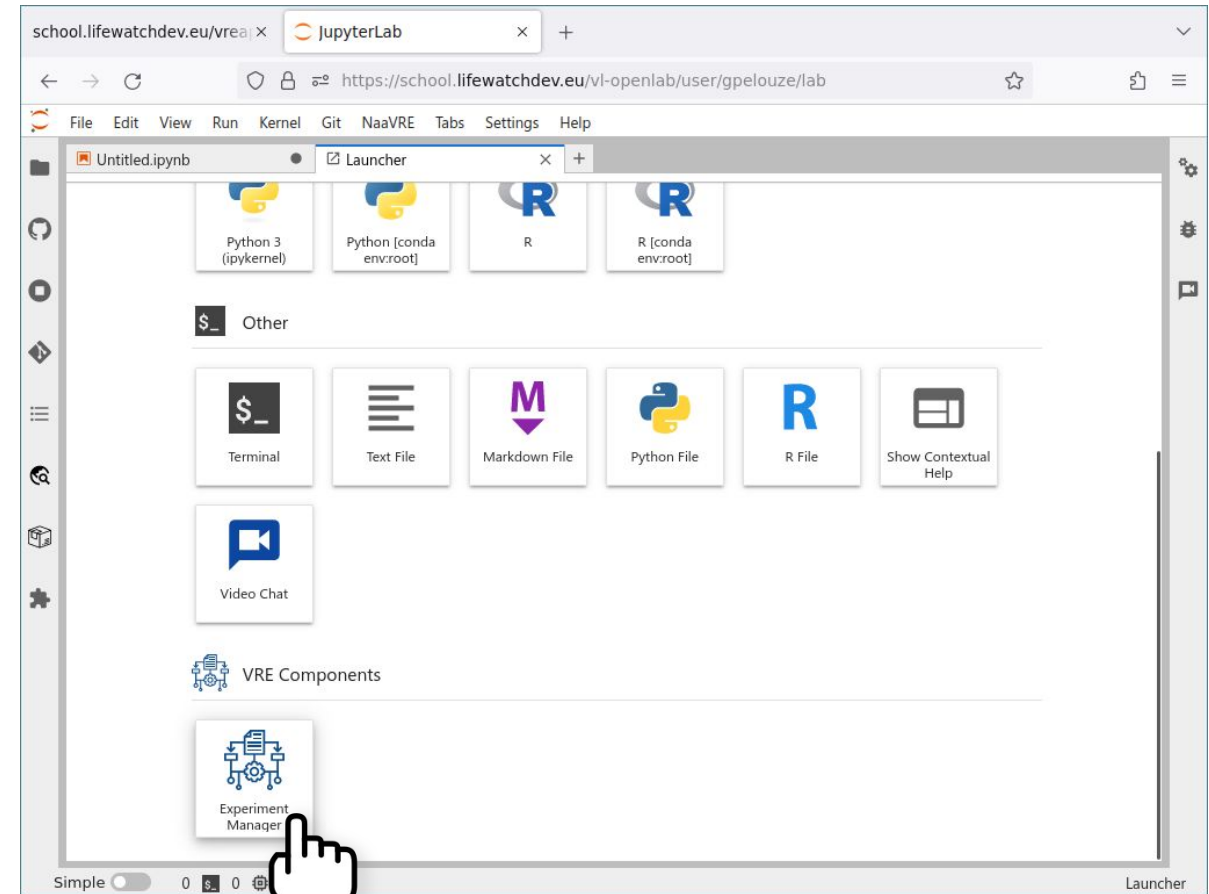
Create a workflow

1. Open the experiment manager



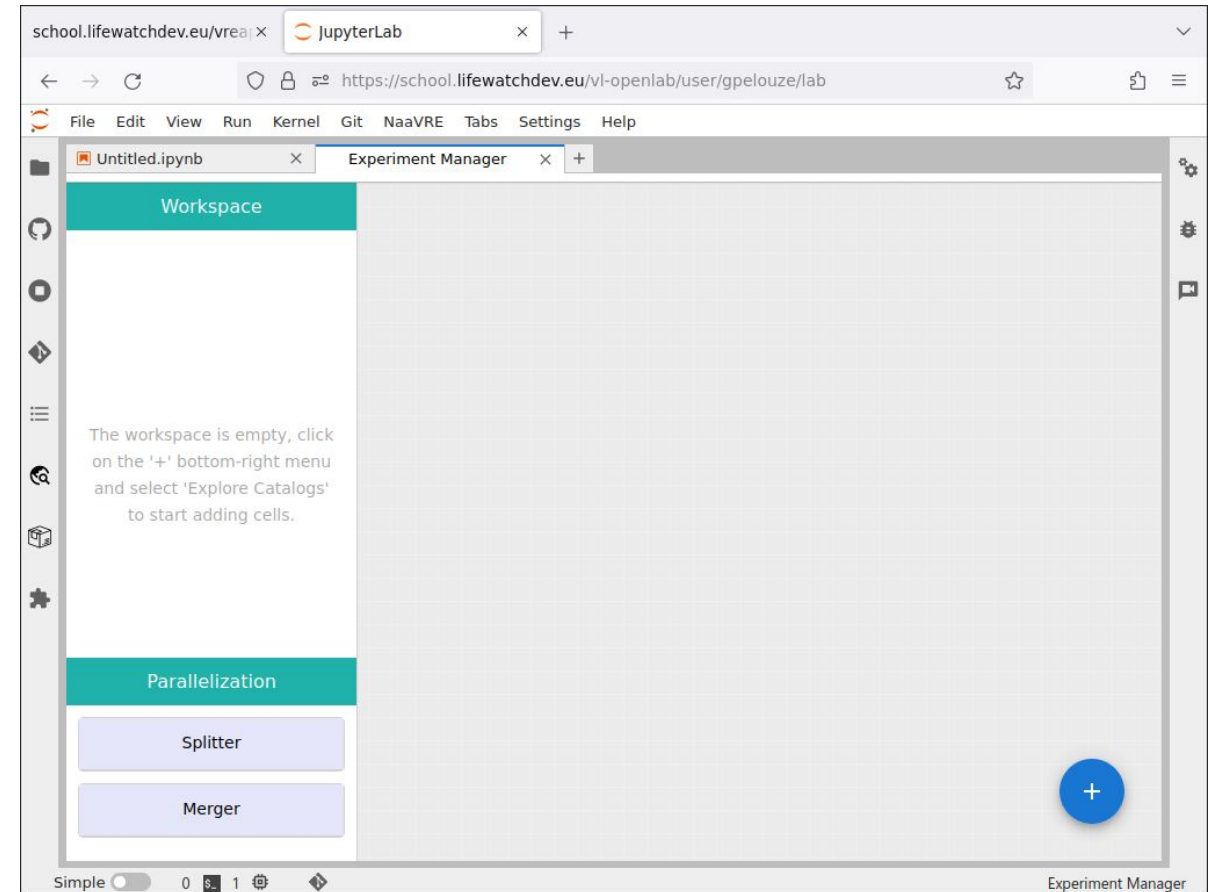
Create a workflow

1. Open the experiment manager



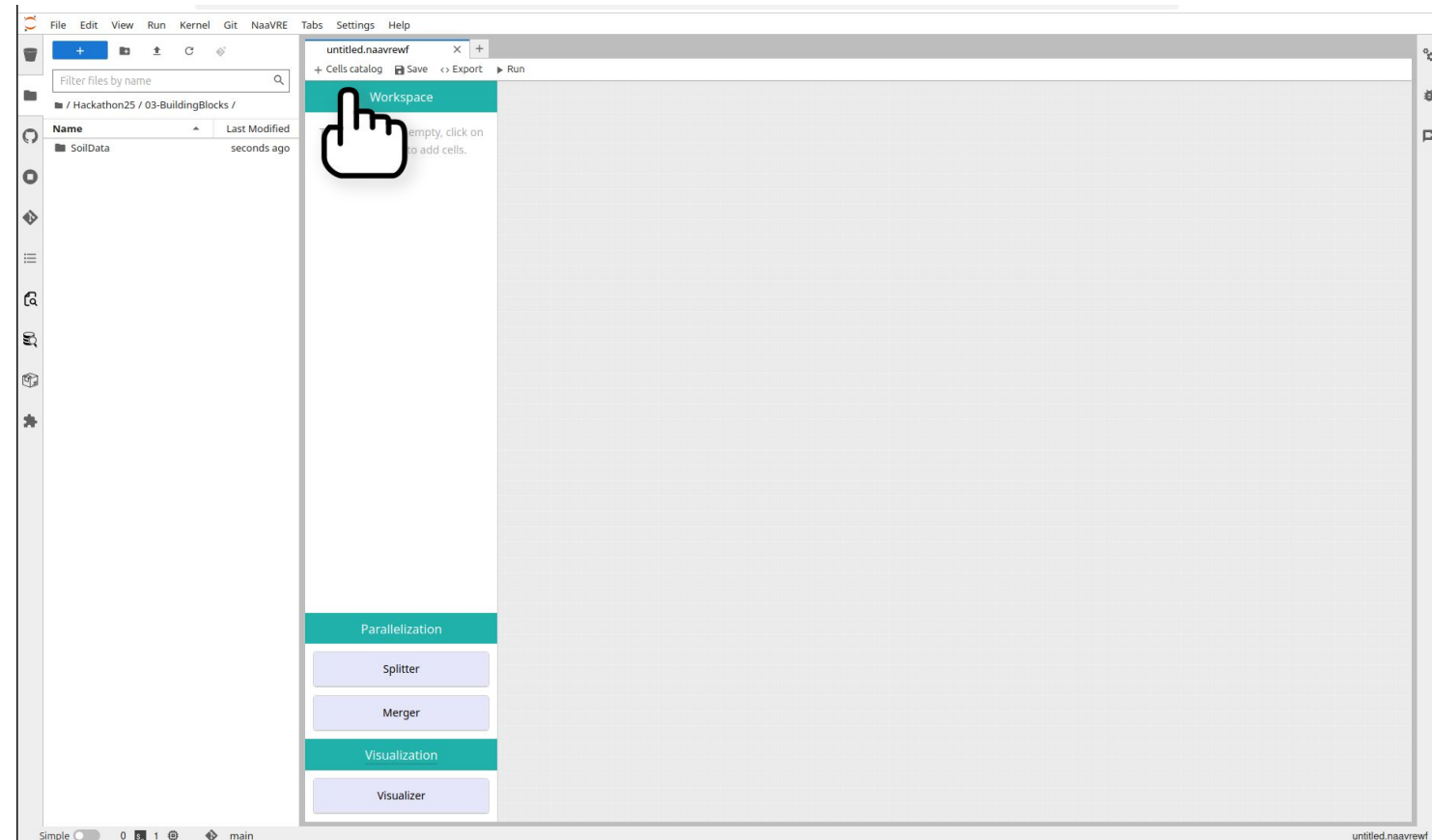
Create a workflow

1. Open the experiment manager



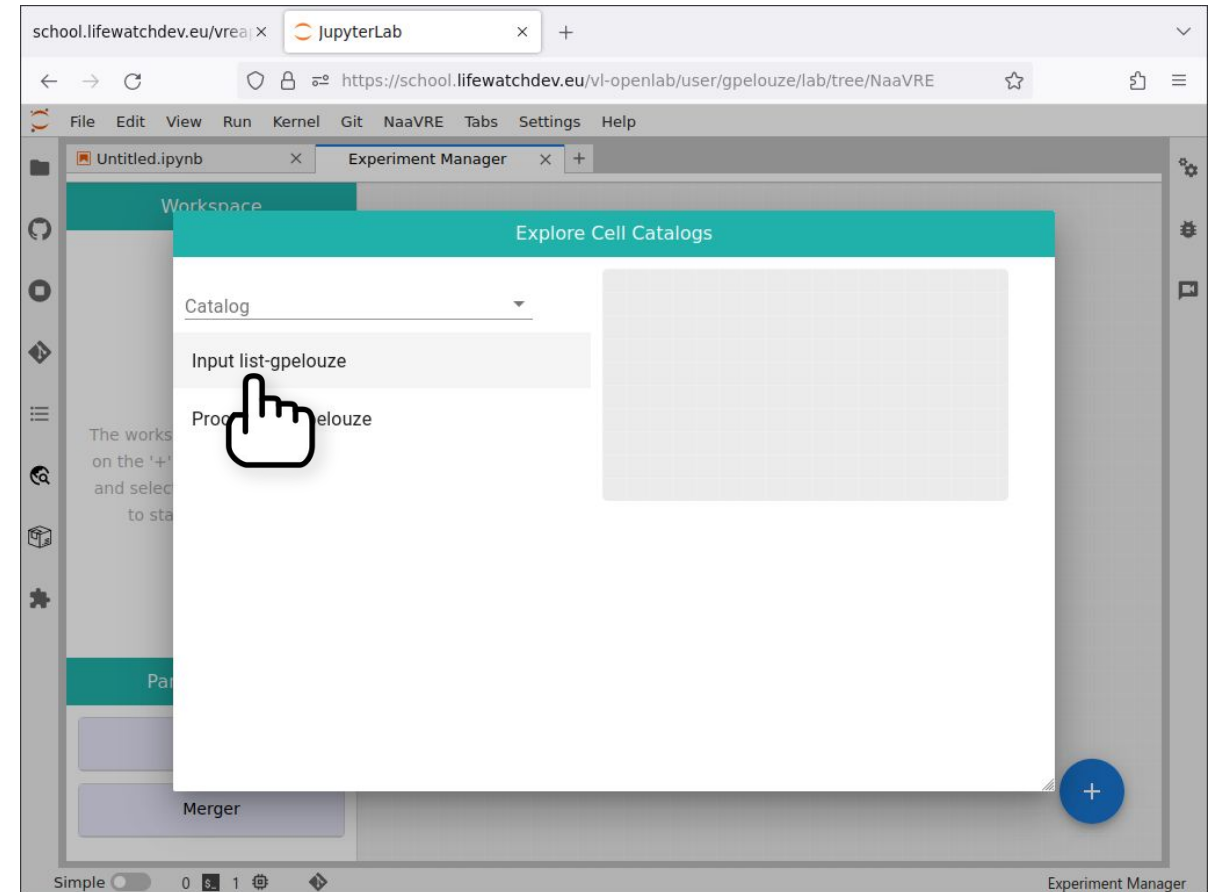
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs



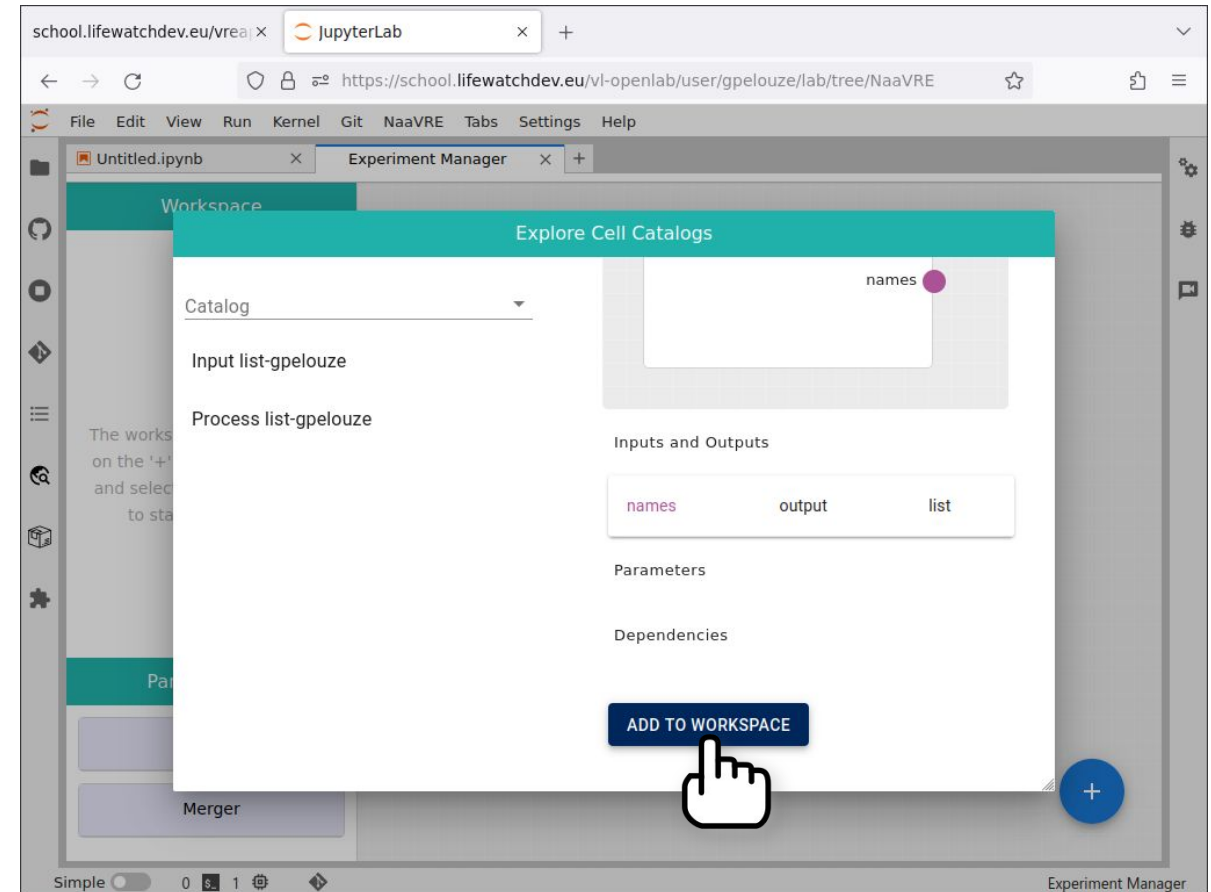
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell



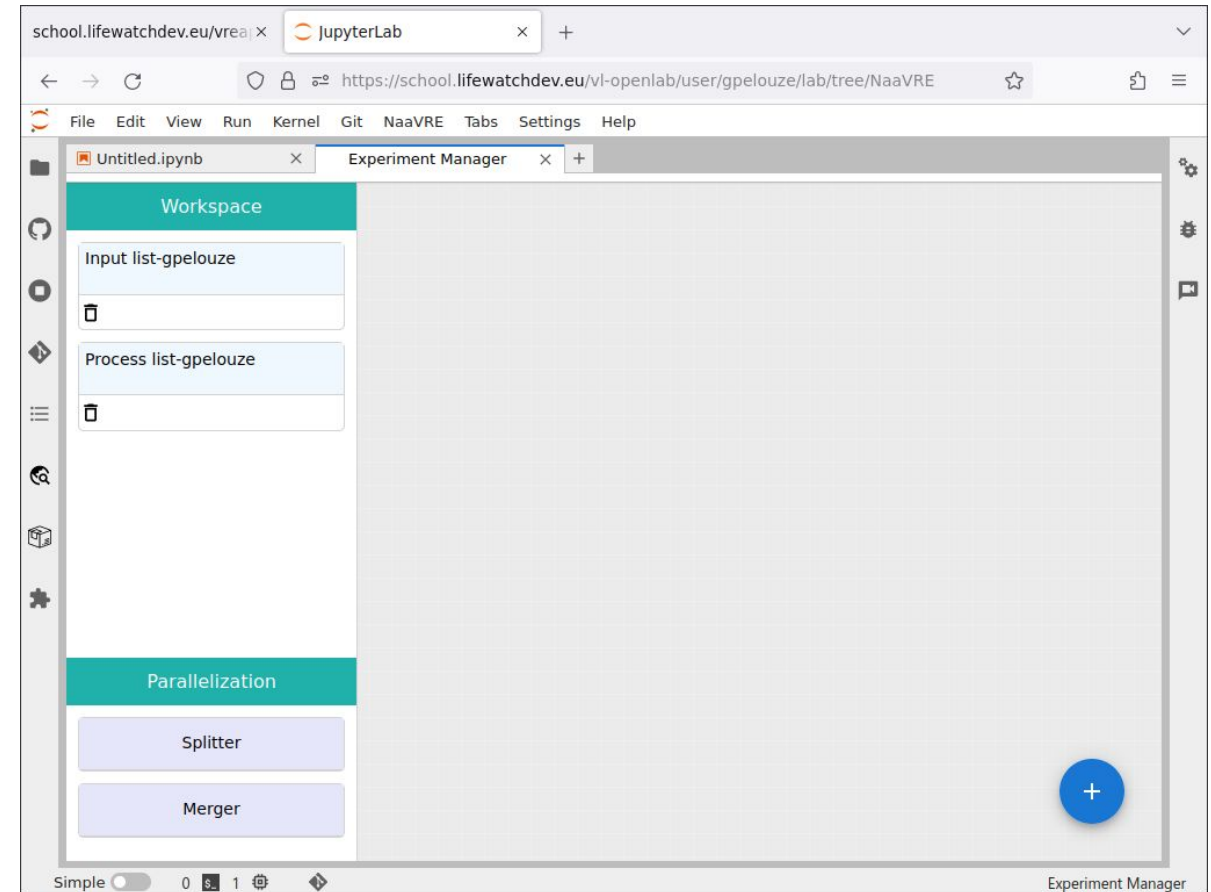
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”



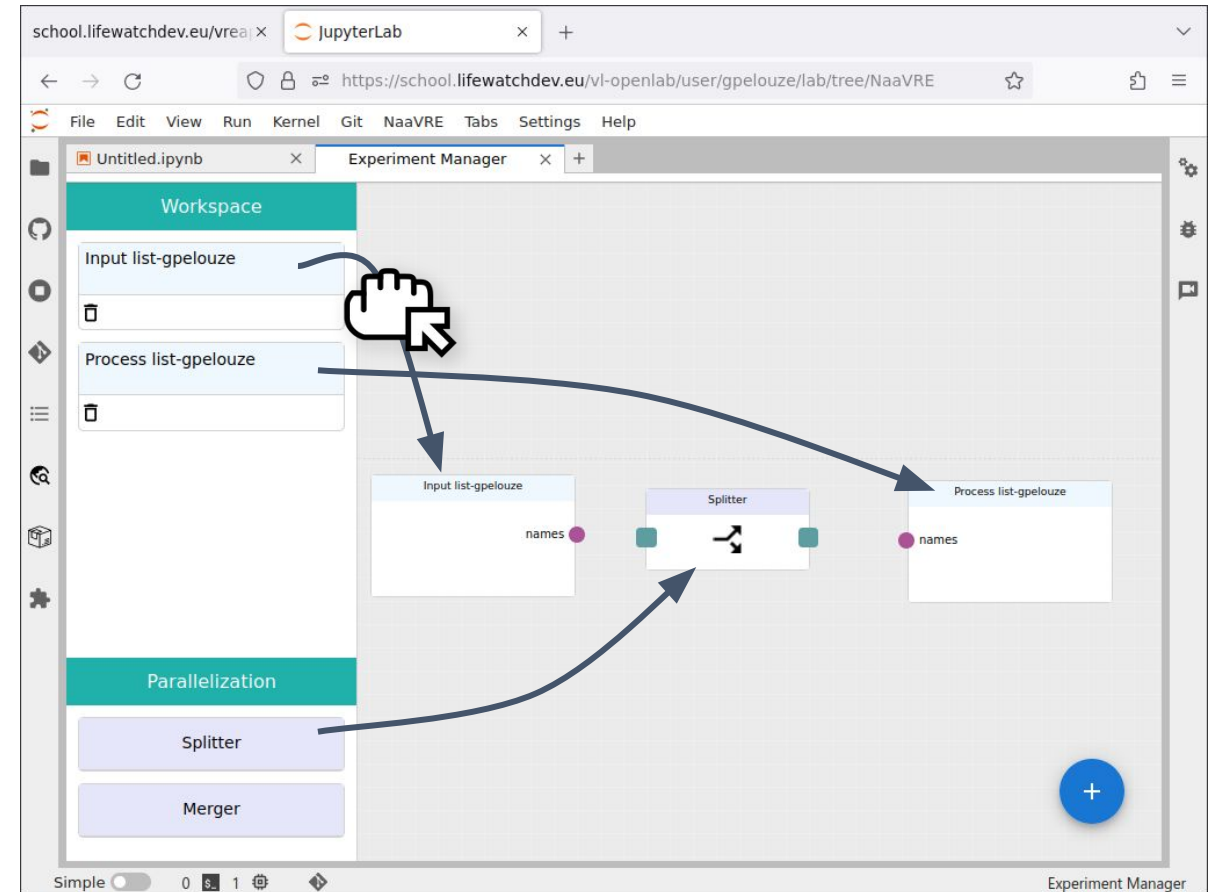
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”
 - d. repeat b–c for other cells



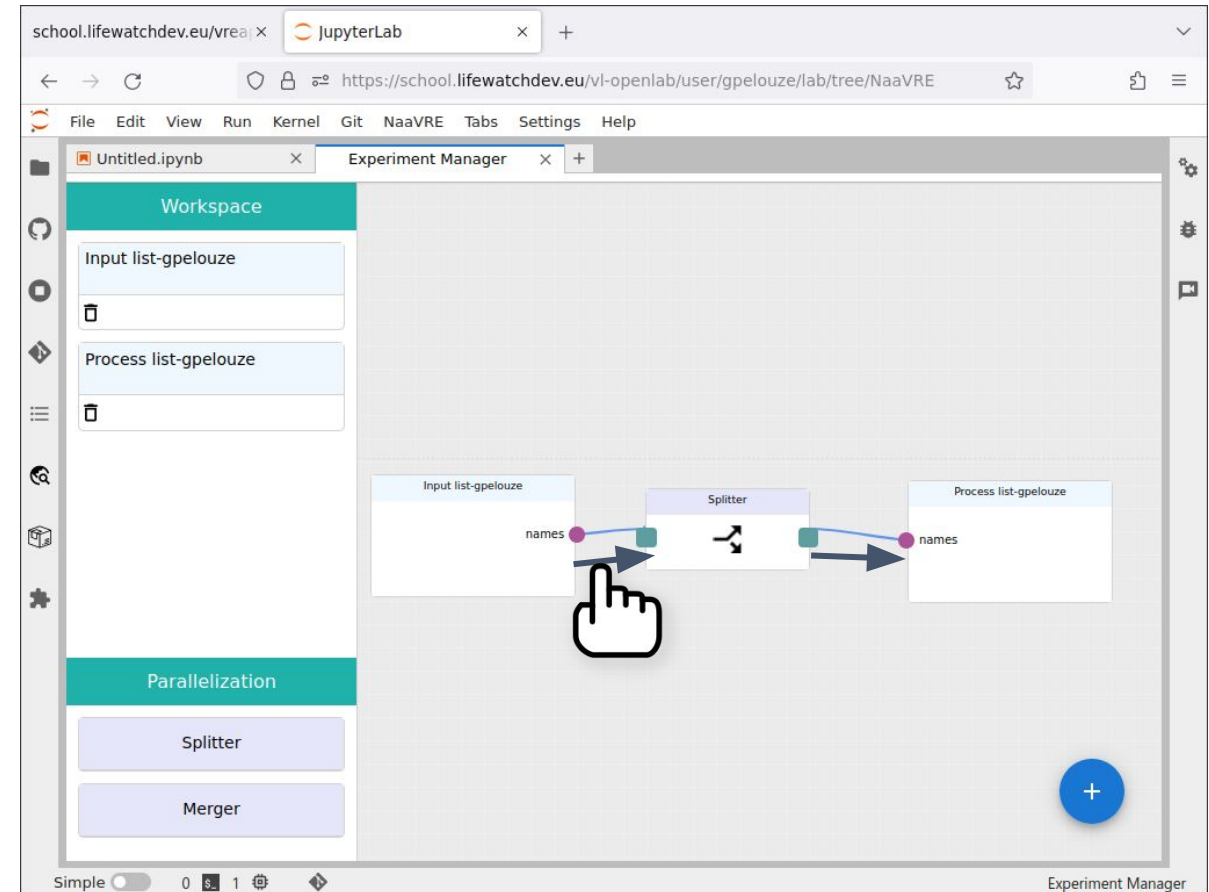
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”
 - d. repeat b–c for other cells
3. Drag and drop the cells onto the canvas



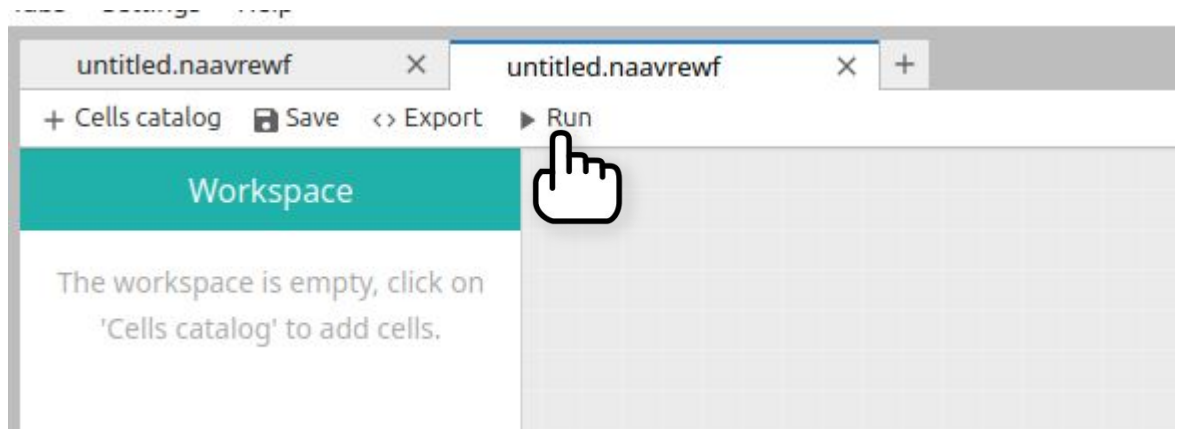
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”
 - d. repeat b–c for other cells
3. Drag and drop the cells onto the canvas
4. Connect the cells



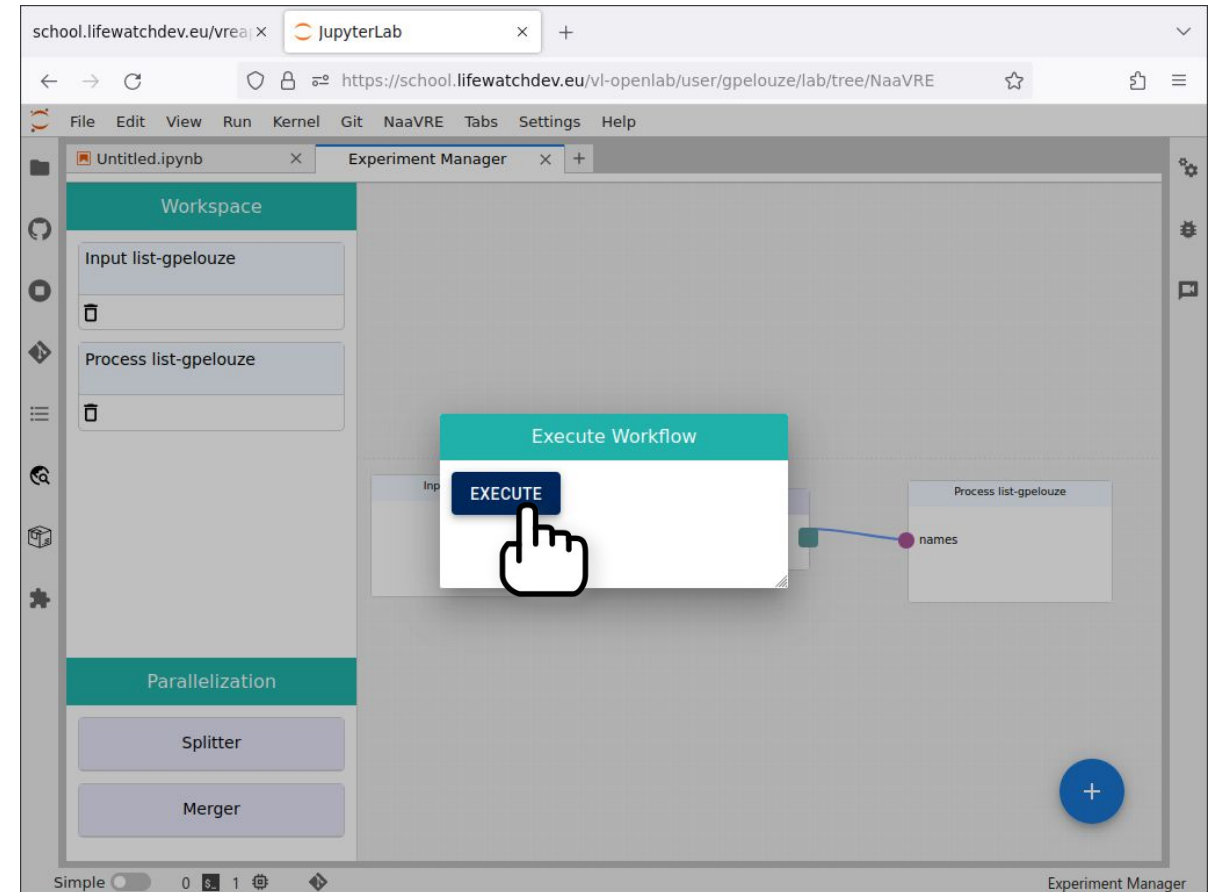
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”
 - d. repeat b–c for other cells
3. Drag and drop the cells onto the canvas
4. Connect the cells
5. Execute the workflow



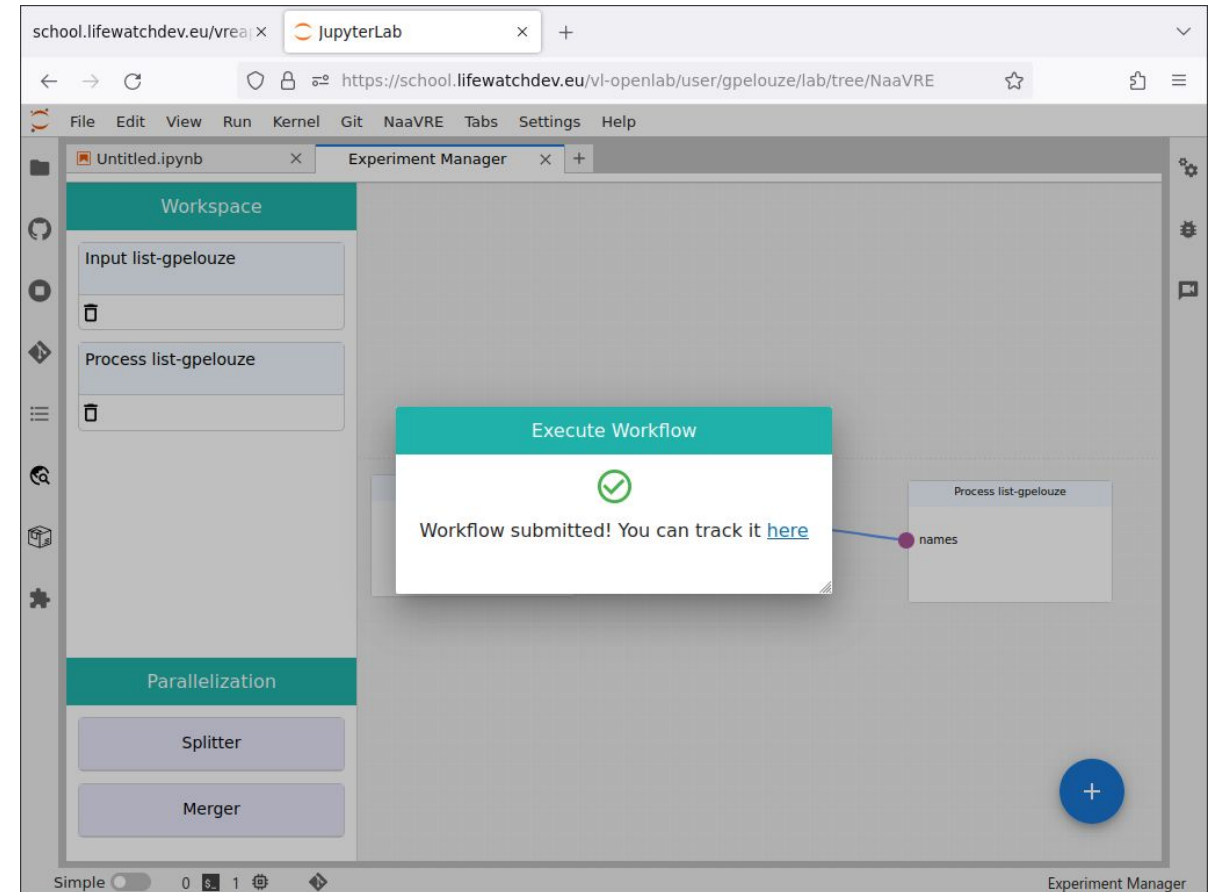
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”
 - d. repeat b–c for other cells
3. Drag and drop the cells onto the canvas
4. Connect the cells
5. Execute the workflow



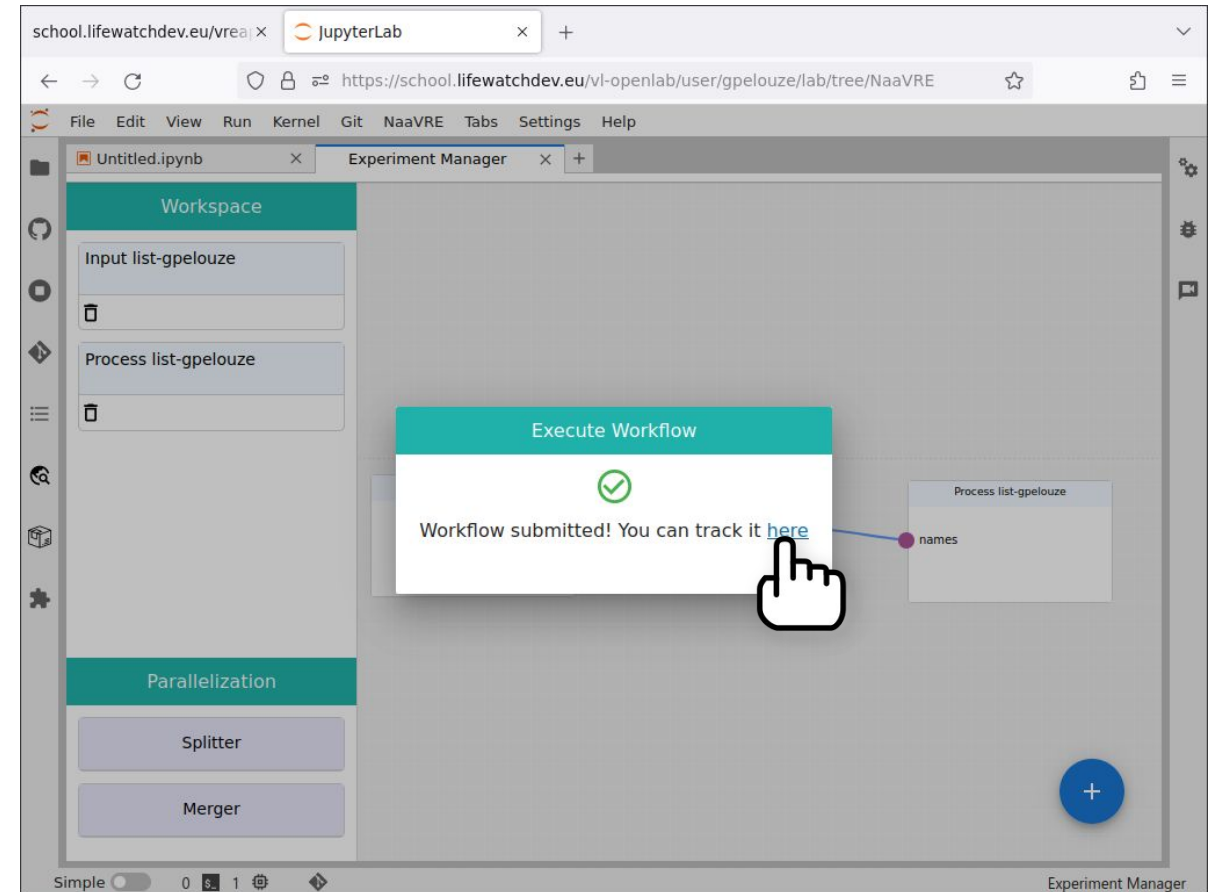
Create a workflow

1. Open the experiment manager
2. Add cells to the workspace
 - a. open the Cells Catalogs
 - b. select a cell
 - c. click on “Add to workspace”
 - d. repeat b–c for other cells
3. Drag and drop the cells onto the canvas
4. Connect the cells
5. Execute the workflow
6. Well done!



Create a workflow

1. Click on the link to open the workflow manager, argo
2. If needed, log in to argo



Create a workflow

1. Click on the link to open the workflow manager, argo
2. If needed, log in to argo
3. Check your workflow status

