MyLCA

A library for generic, parametric and modular LCA

Maneval, Victor
Perdu, Fabien
Jamet, Didier
Goutelle, Mathieu
Maillot, Isabelle

Aalborg, 23rd September 2026

LCA practices at CEA-LITEN

"Developing differentiating energy technologies for carbon neutrality, transferable to industry"



Photovoltaic



Battery



Hydrogen and sustainable fuels



Smart grid, efficient energy systems



Circular economy and materials

LCA

LCA experts
×
Technical experts

Evaluation	*"what are the impacts?"*
Eco-design	*"how to minimize them?"*
Eco-innovation	*"what research roadmap?"*

Our needs

Advanced LCA

- Parametric,
- Prospective,
- Dynamic,
- etc.

Collaboration & Capitalization

- Collaborative edition
- **Share & reuse LCA-model blocks**

Usability

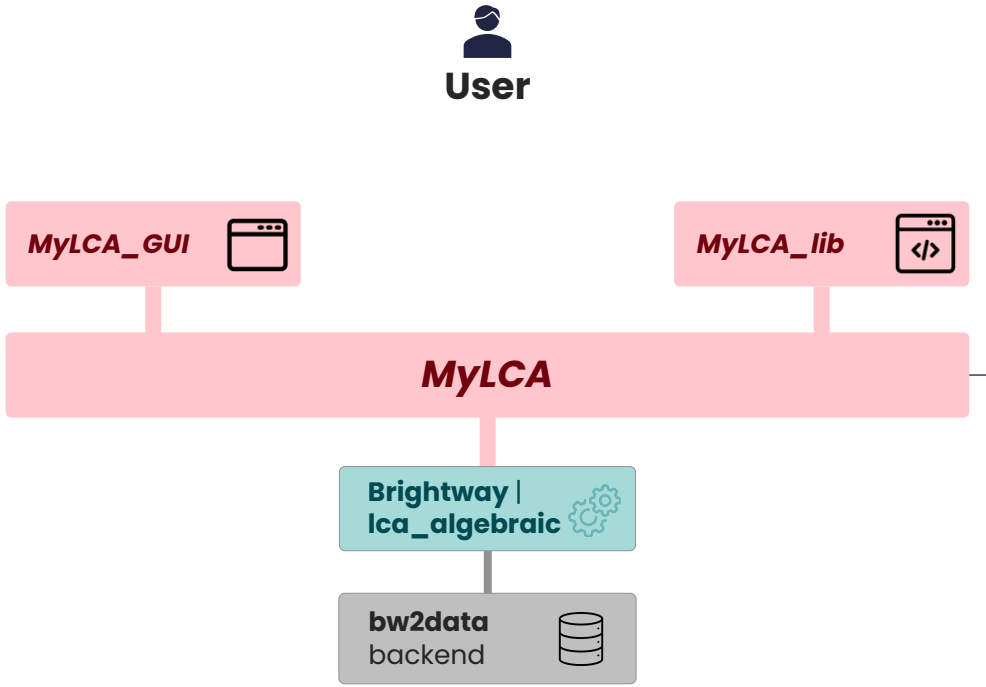
- Code-friendly & code-avert users

Interoperability

- LCA and other simulation tools

MyLCA, in brief

The goal



- Parametric reusable MyLCA models, **JSON** format
- Calculation tasks for Brightway

An **overlayer to Brightway**, to define **parametric**, **reusable** and **shareable LCA models**, open to a **GUI** as much as to **Python** | Open-source.

Workshop program

1. **The Brightway ecosystem**

   Your usage and the limitations you face (interactive session Mentimeter)

2. **MyLCA concepts, through a use case**

3. **Feedback gathering** (interactive session Mentimeter)

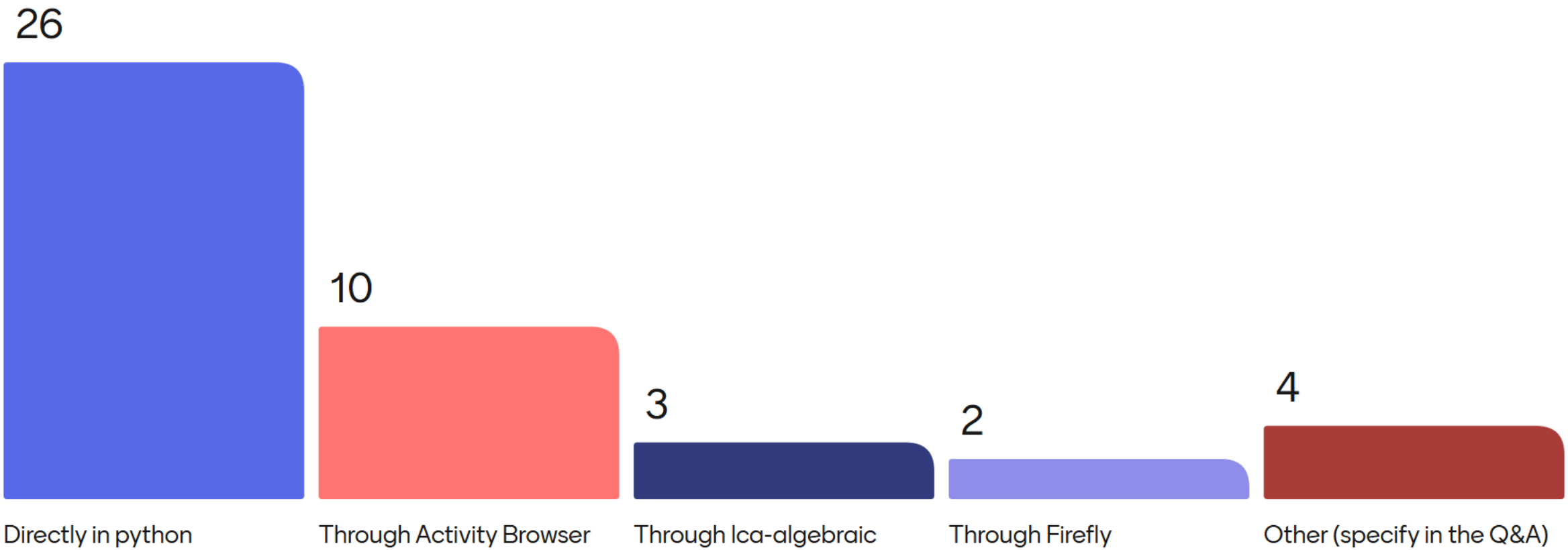
1. The Brightway ecosystem

Usage and limitations

Join at menti.com | use code 4597 2211

Mentimeter

How do you use Brightway?



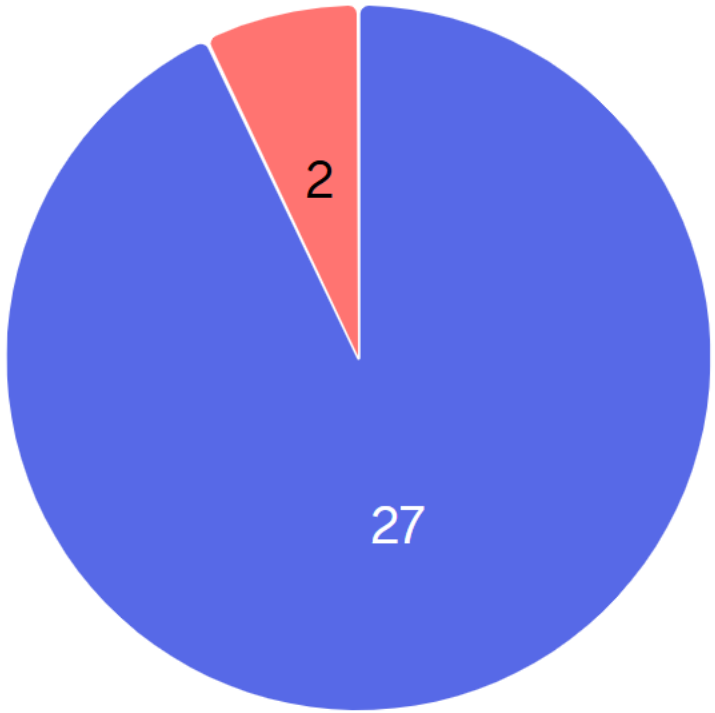
Show trends D

1 2 28 / 34

Join at menti.com | use code 4597 2211

Mentimeter

Where is your Brightway installed?



27	Locally on my computer
2	Centralized on a shared server

Show trends D

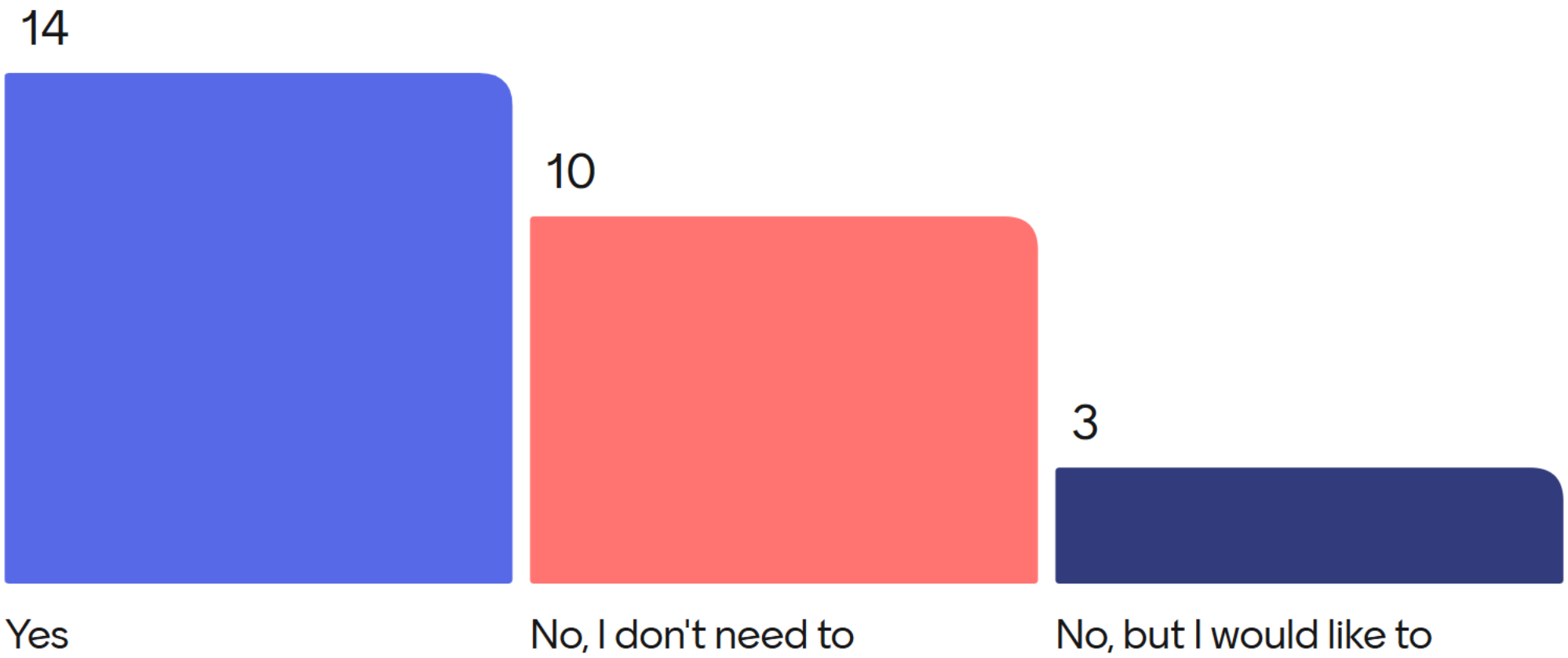
2

29 / 34

Join at menti.com | use code 4597 2211

Mentimeter

Do you **share** your Brightway projects/databases/inventories with colleagues?



Show trends D

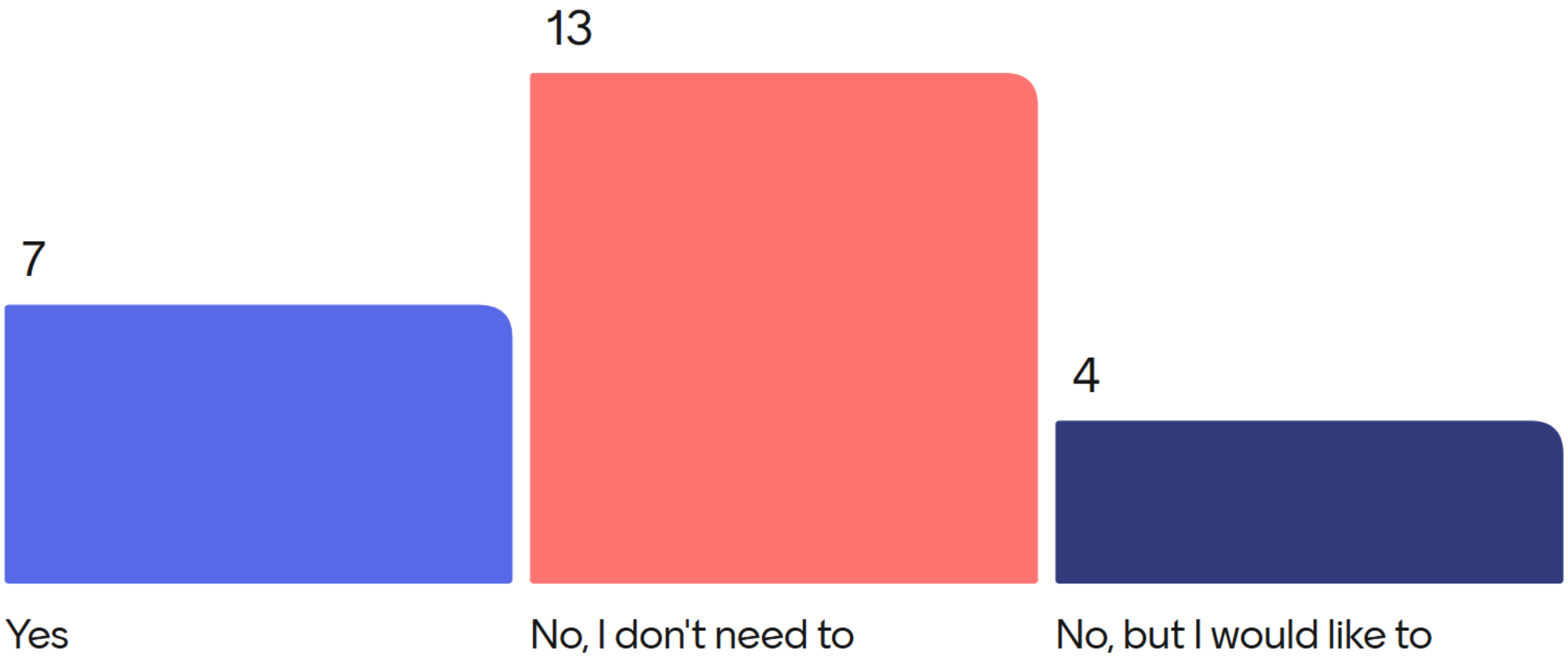
2

27 / 34

Join at menti.com | use code 4597 2211

Mentimeter

Do you **collaborate** on your Brightway projects/databases/inventories with colleagues?



Show trends D

2

24 / 34

Join at menti.com | use code 4597 2211

Mentimeter

If yes, do you have **convenient ways** of doing it?

Not really	ElecImpact ;)	firefly
firefly	github	Gilab
I made a silly JSON format that I use to reconstruct projects	Share the Excel output in OneDrive	Excel

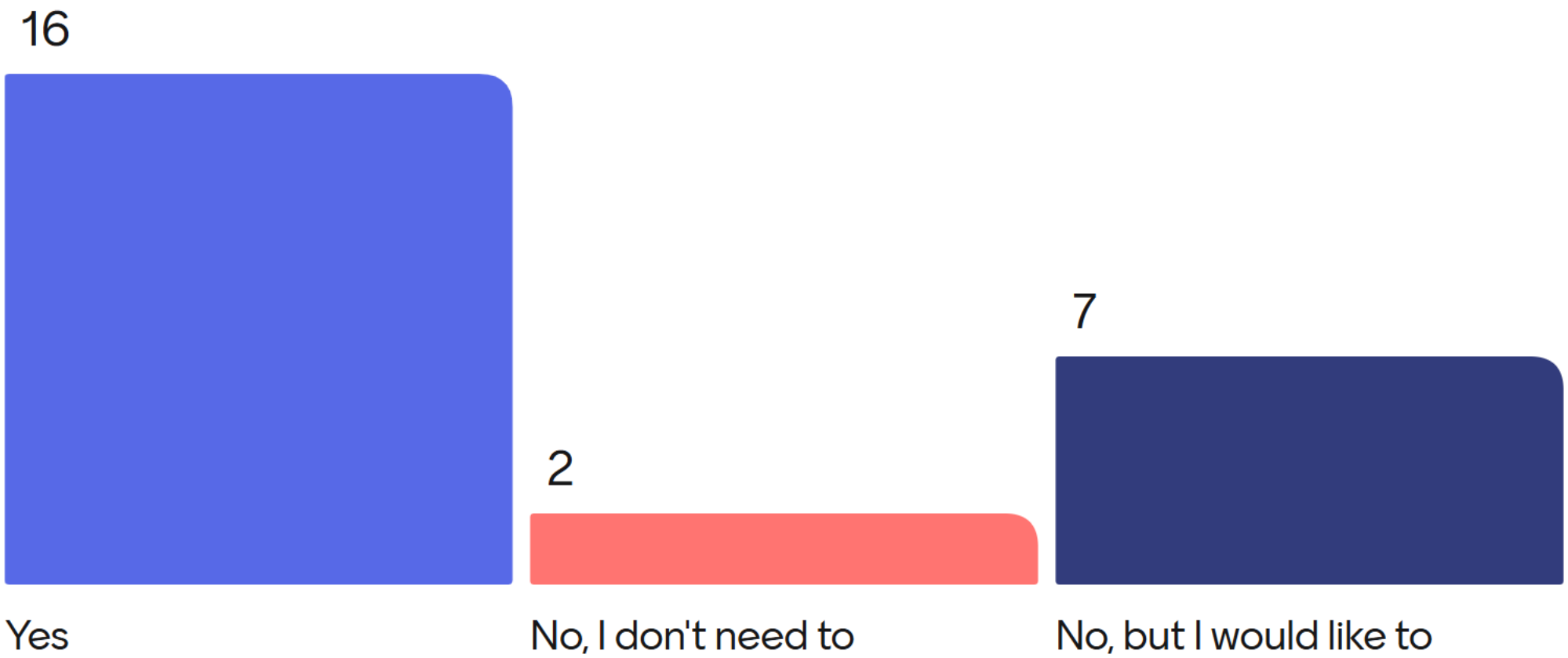
2

8 / 34

Join at menti.com | use code 4597 2211

Mentimeter

Do you do **parametrised LCA models** and **calculations**?



Show trends D

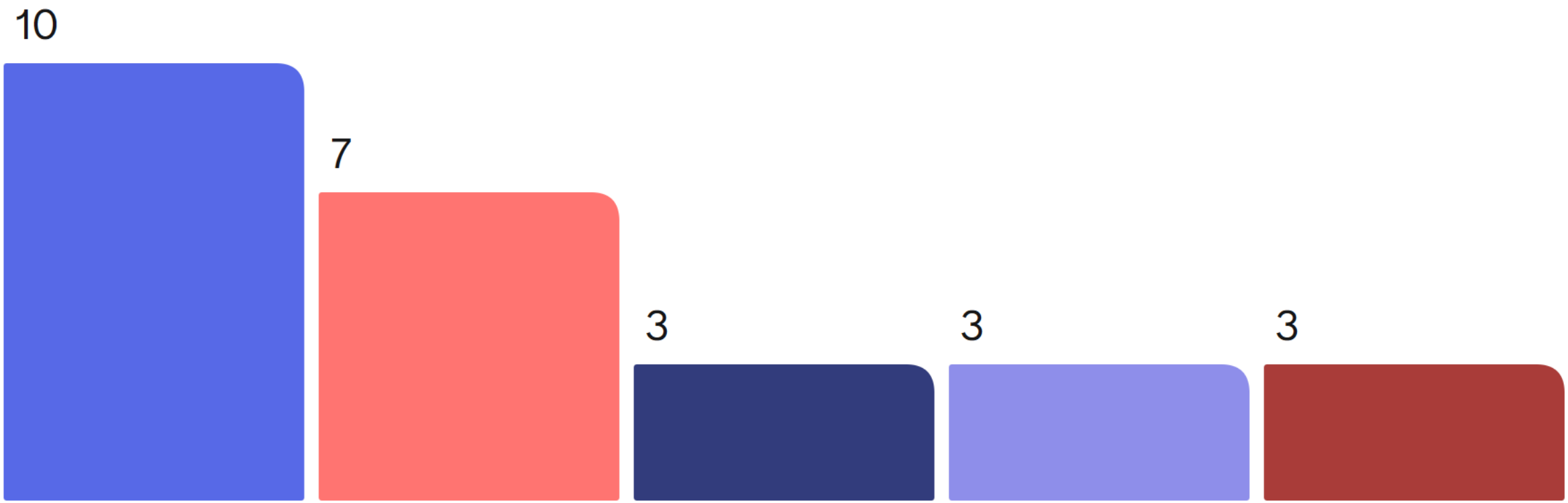
2

25 / 34

Join at menti.com | use code 4597 2211

Mentimeter

If yes, how?



Show trends D

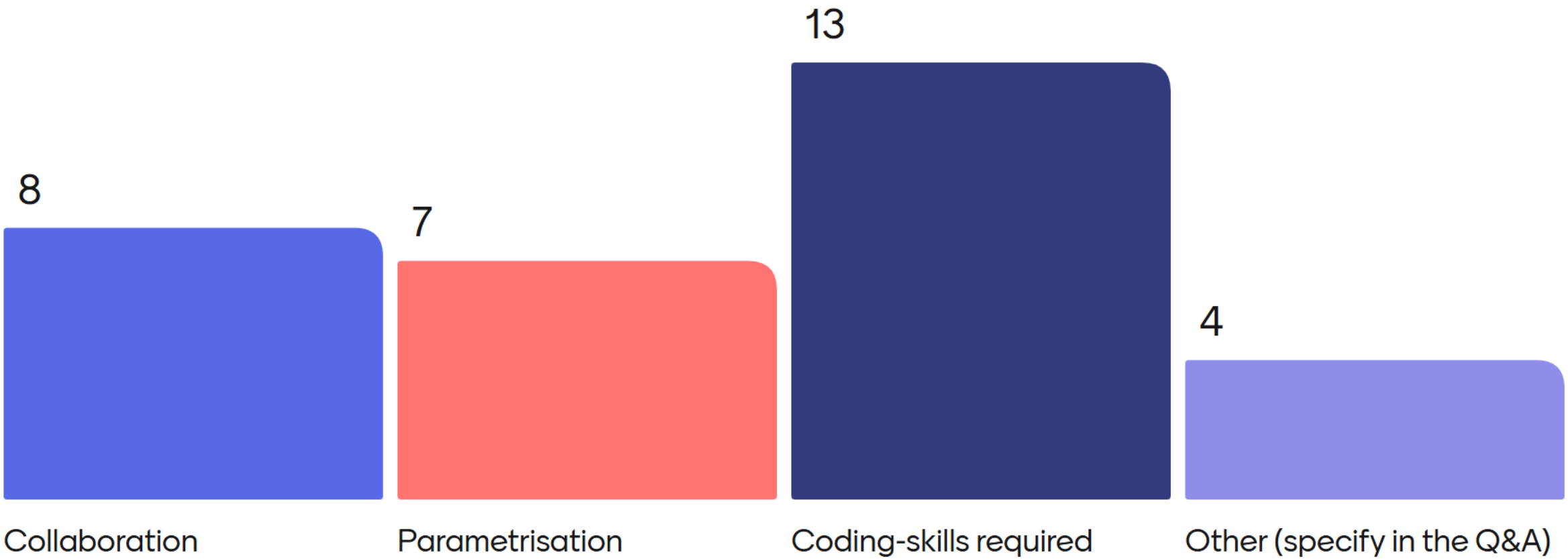
2

21 / 34

Join at menti.com | use code 4597 2211

Mentimeter

What are the **biggest limitations** you encouter with the current Brightway ecosystem tools ?



Show trends D

2

23 / 34

2. MyLCA concepts

A battery use case

A simple use case

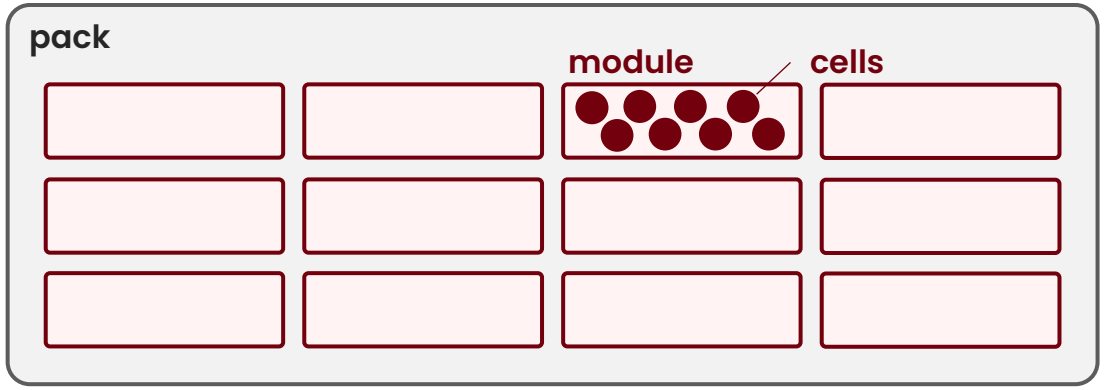
A medium-size EV battery pack

Functional unit: 40 kWh battery pack

- 12 modules,
- 8 cells each,

Scope: cradle to gate

Indicator: GWP100



Identical cells: model one cell, one module taking 8 cells, and one pack taking 12 modules.
Just set amounts of exchanges.

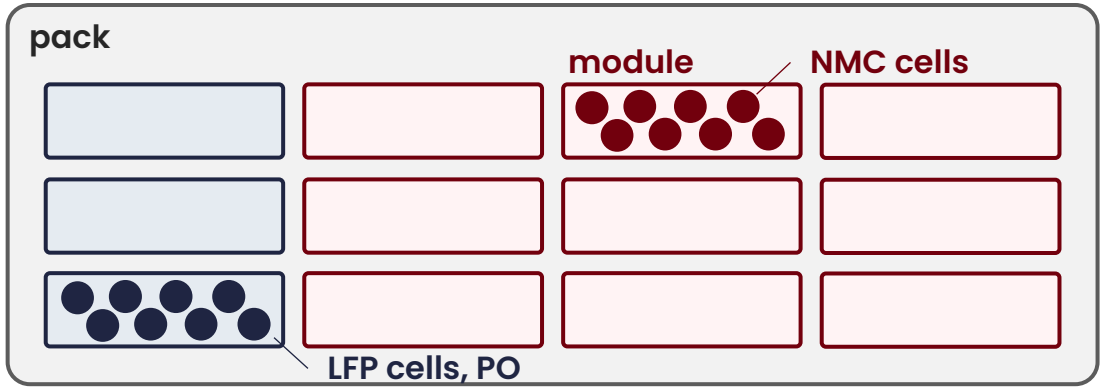
The hiccup

Same cell specifications, but two references?

Two cell references:

- **NMC cells**, from South Korea
- **LFP cells**, from Poland

Different cells, from different production routes,
but cells with the same design and requirements.



Two instances of **one shared cell design**, wired into two different manufacturing contexts?

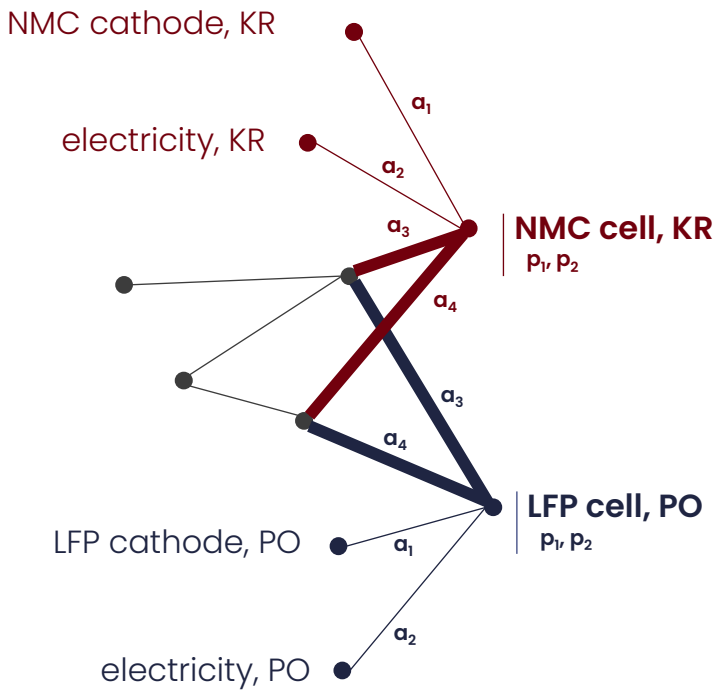
The diagnosis

Exchanges are **casted references**.

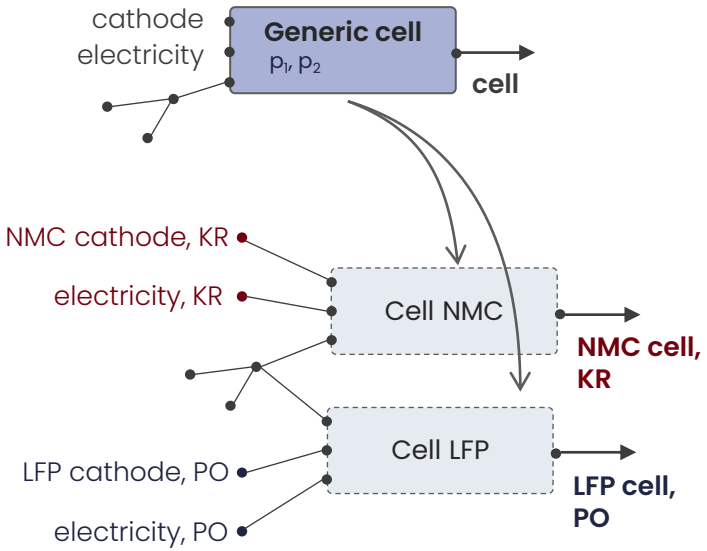
"One design - two contexts" means two hand-authored activities, and **duplicated parameters, exchanges**, and **amounts**.

We miss a notion of "same recipe – wired differently".

what we have



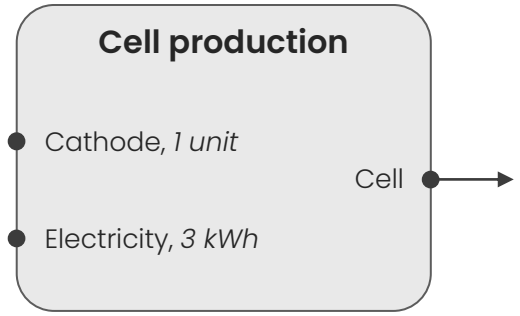
what's missing



In Brightway, **activities are fully wired** – no separate layer for the "recipe"

A socket, not a wire

First piece: **ports**



A port is a **typed connection point**, without a specific source required yet.

The port contains at least the input name and the required amount.

The **dataset becomes a function**: accepting **different inputs**, **reusable** and **shareable**.

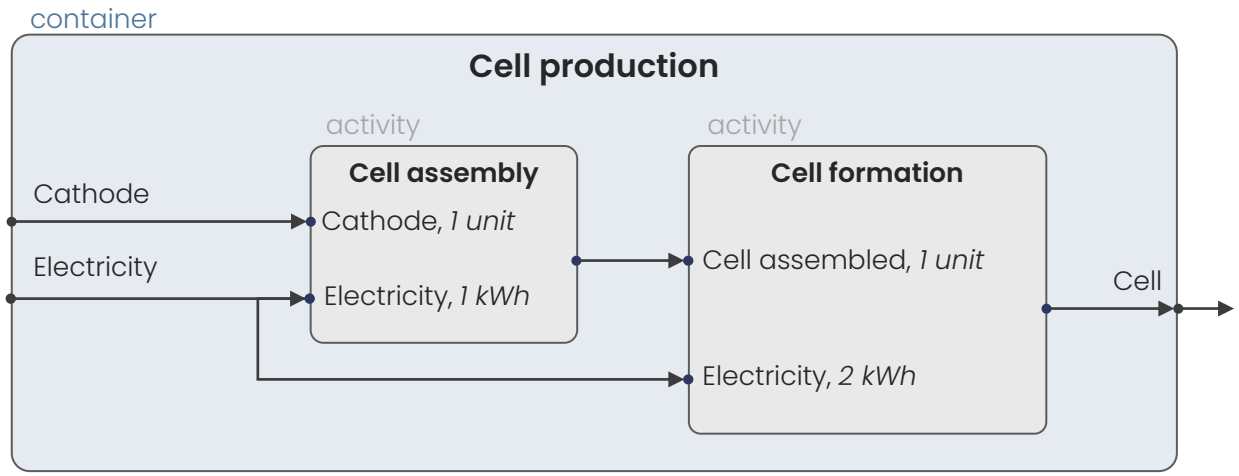
```
cell_prod = my_project.new_activity(<…>)

cell_prod.new_port("cathode", direction="in",  amount=1)
cell_prod.new_port("electricity", direction="in", amount=3, unit="kWh")
cell_prod.new_port("cell", direction="out« , amount=1)
```

Design the cell once, use it in **different contexts**, **share it with other** users.

On a plug-in box

Second piece: **containers**



Not a simple dataset: **a container can enclose more complexity**.

Say “cell production is composed of different successive steps" which can be grouped in a container as a sub-model.

A container also has ports.

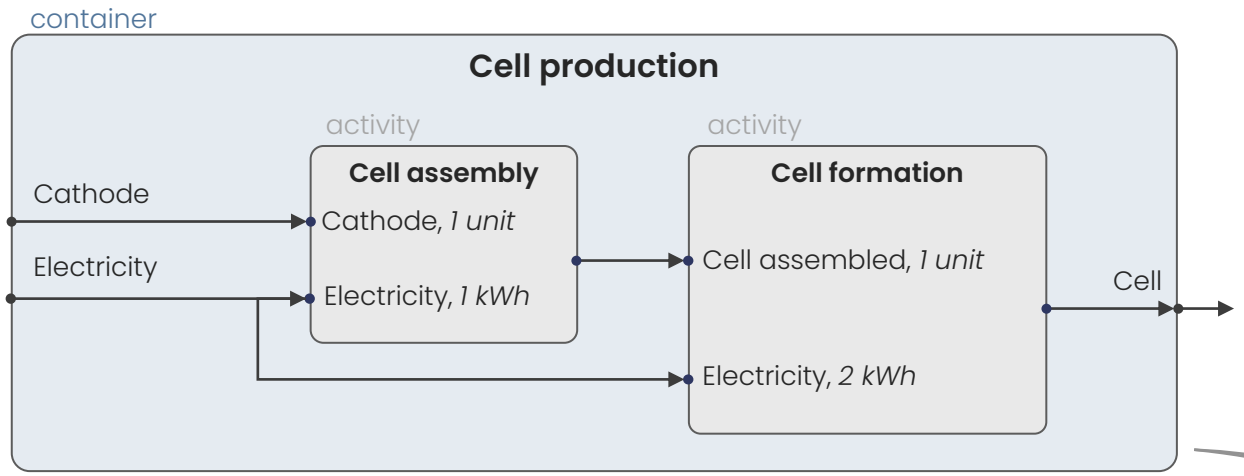
```
cell_prod = my_project.new_container(<…>)

cell_prod.new_activity(cell_assembly)
cell_prod.new_activity(cell_formation)
…
```

Define **complex sub-models, as functions** of their inputs, **reuse them** and **share them**.

On a reusable plug-in box

Third piece: **instantiation of templates**



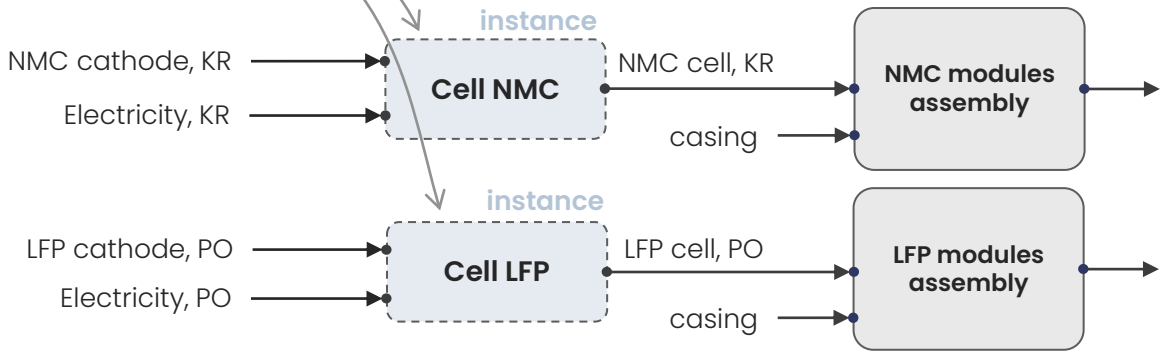

```
cell_prod = my_project.new_container(<…>)

my_project.new_template(cell_prod)

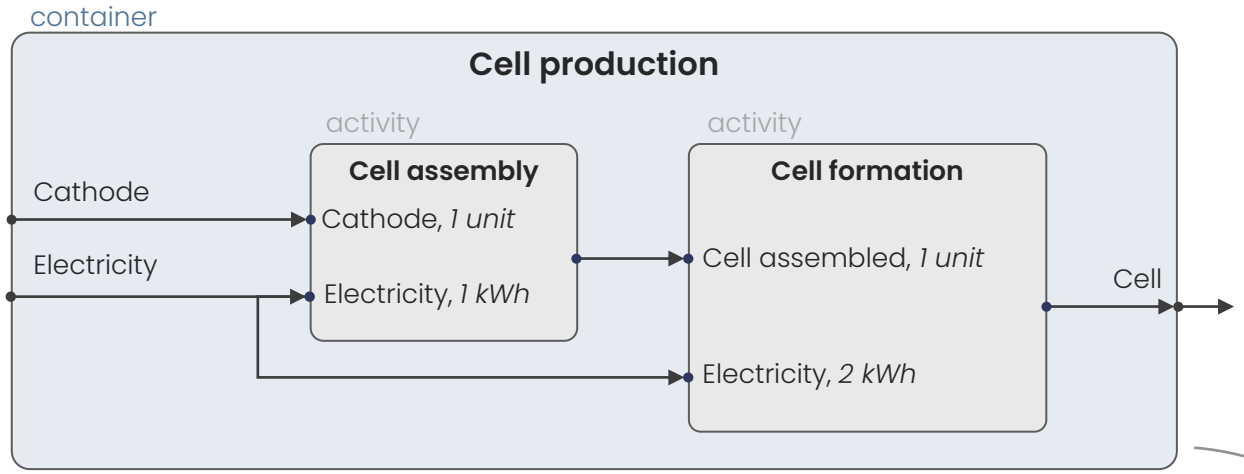
nmc = my_project.new_instance(cell_prod, 'Cell NMC', …)
lfp = my_project.new_instance(cell_prod, 'Cell LFP', …)
```

The **design** is **defined once, in a container**.

The container can **serve as a template** and be **instantiated** and **used** at **different places** in a model, **taking different inputs**.



Templates and instances can be nested



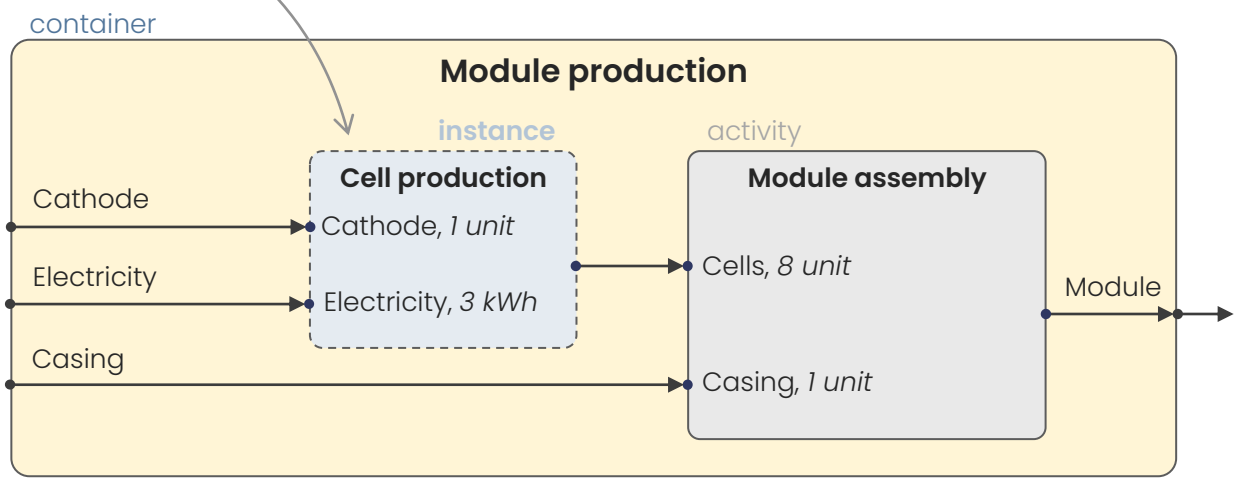

```
cell_prod = my_project.new_container(<…>)
my_project.new_template(cell_prod)

module_prod = my_project.new_container(<…>)
module_prod.new_instance(cell_prod, …)
```

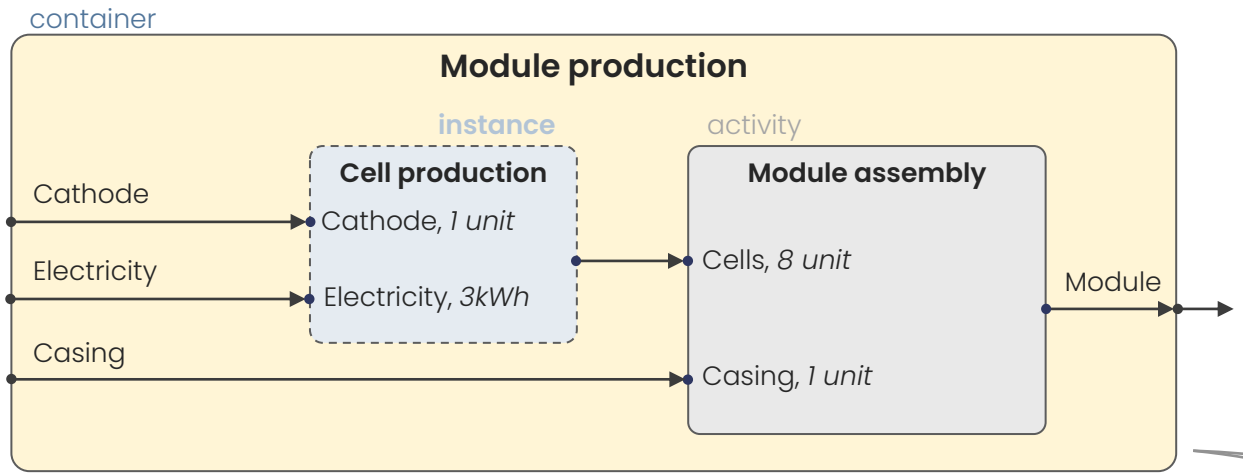
An instance of the "**Cell production**" template is nested inside the "**Module production**" container, which can then be used as a bigger template.

Containers **from any author** can be **aggregated, used and reused** across a model, and shared.

⇒ **Reusable LCA-model blocks**



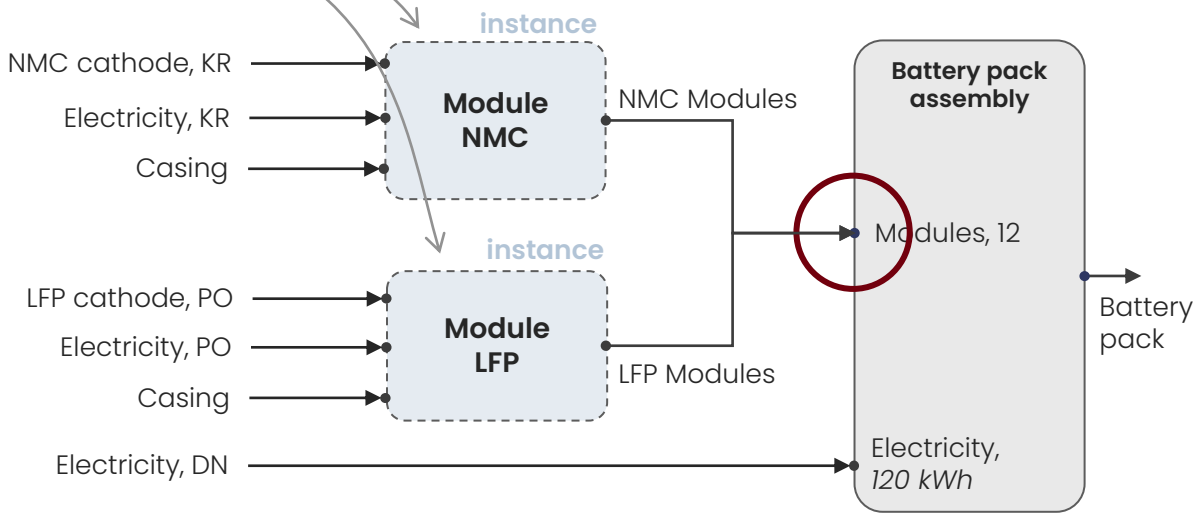
Ports can serve as mixers




```
pack_assembly.ports['modules'].amount = 12
pack_assembly.ports['modules'].exchanges = [
            {source: module_NMC, ratio: 0.75},
            {source: module_LFP, ratio: 0.25}]
```

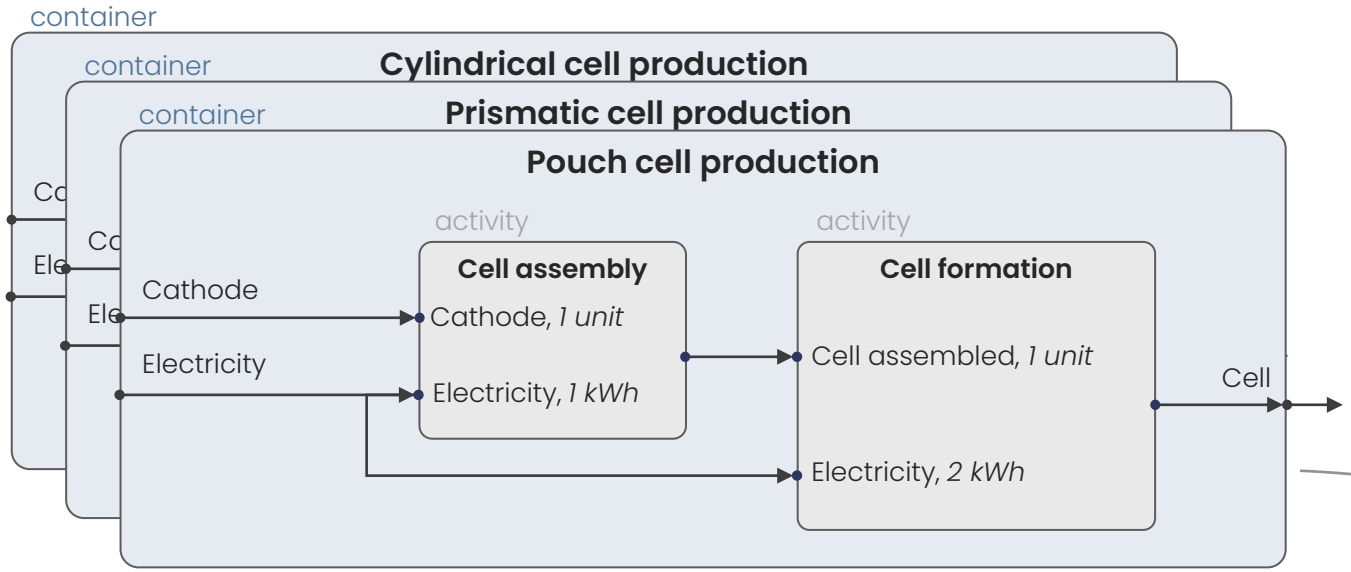
Ports of activities and containers **can have several source exchanges**.

A **ratio** can be set, to **split the total amount** between each source.

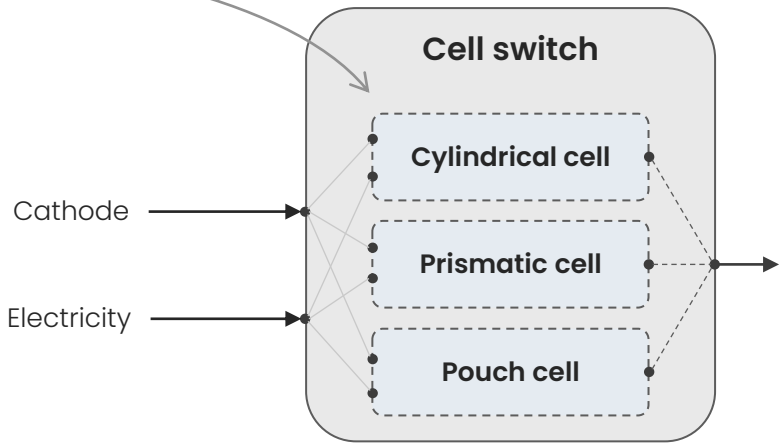


Different cell types?

Forth piece: **switches**




```
cell_pouch = my_project.new_container(<…>)
my_project.new_template(cell_pouch)
…
cell_switch = my_project.new_switch(<…>)
cell_switch.new_instance(cell_pouch, …)
```

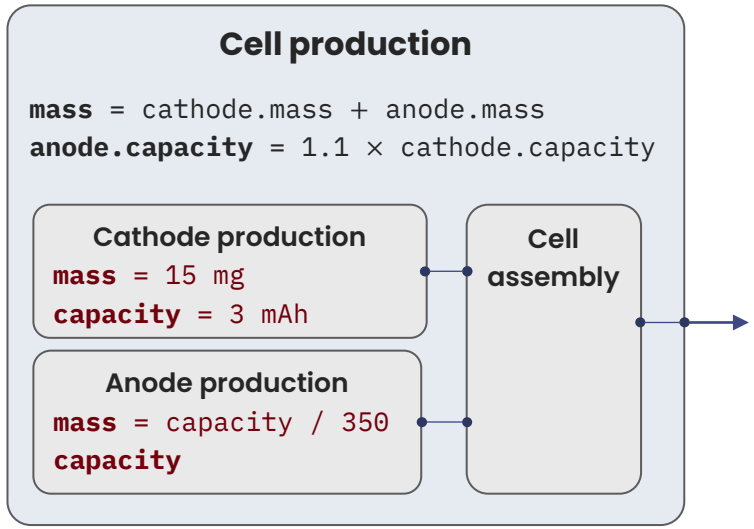


Cells can actually be of **different types**: from the same inputs. Build one template for each, and put instances in a **switch**.

Enables **modularity in the model**, for **scenario** calculations and comparison.

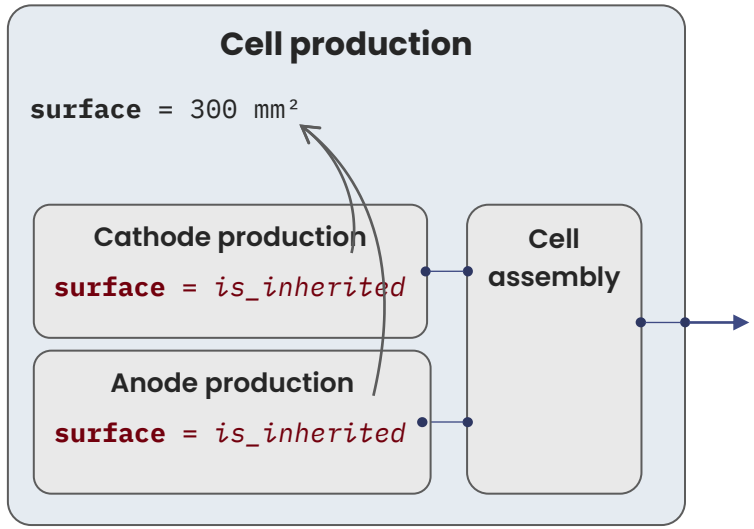
Alternatives in a switch **can be from any author**.

All backed by a parameter system



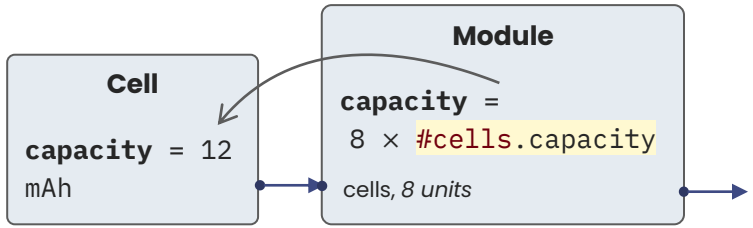
Parameters are **defined within activities** and **containers**, so that sub-models can be self-sufficient.

A **container** can **access its content** in **reading** and **writing**.



To enable « upward » access, a **child** can **inherit a value from its ancestors**.

This enable automatic context-dependant affectation, **whichever the parent is**.



An activity or container can access parameters from its inputs, throught its ports.

This also enables automatic context-dependant affection, **whatever the input is**.

Where are MyLCA's real benefits?

Everything so far is **class/instance modeling**, doable in Python.

Paired with a **GUI**:

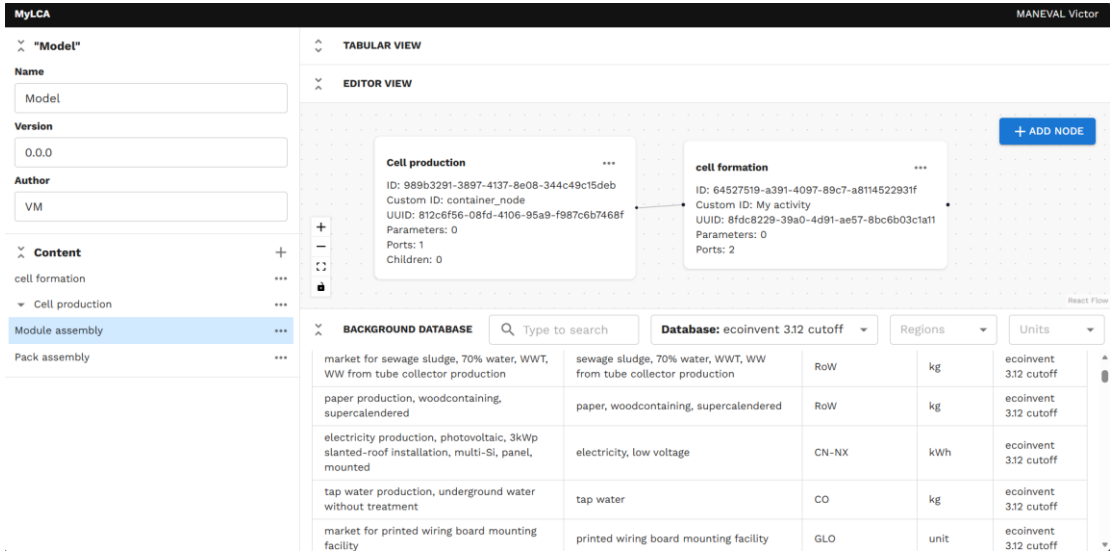
- Drag a *cell* block, connect its ports, pick alternatives from a dropdown, drag a *module* block, etc.

Same functionalities — no script required.

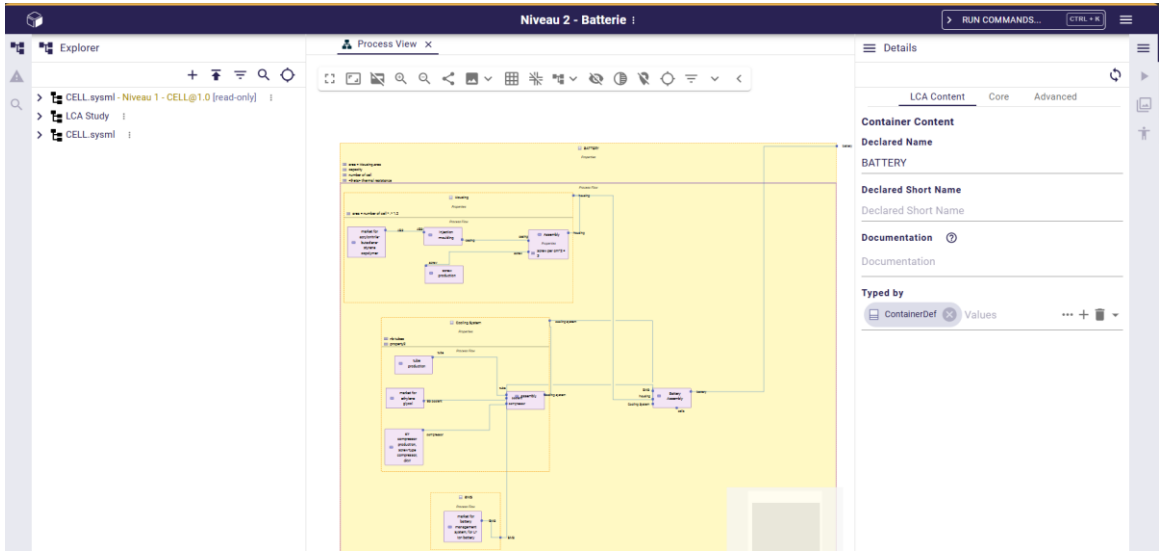
Paired with **Model-Based Systems Engineering**:

- Conduct LCA directly from MBSE system models and editors

Same system – described only once.



LCA web GUI



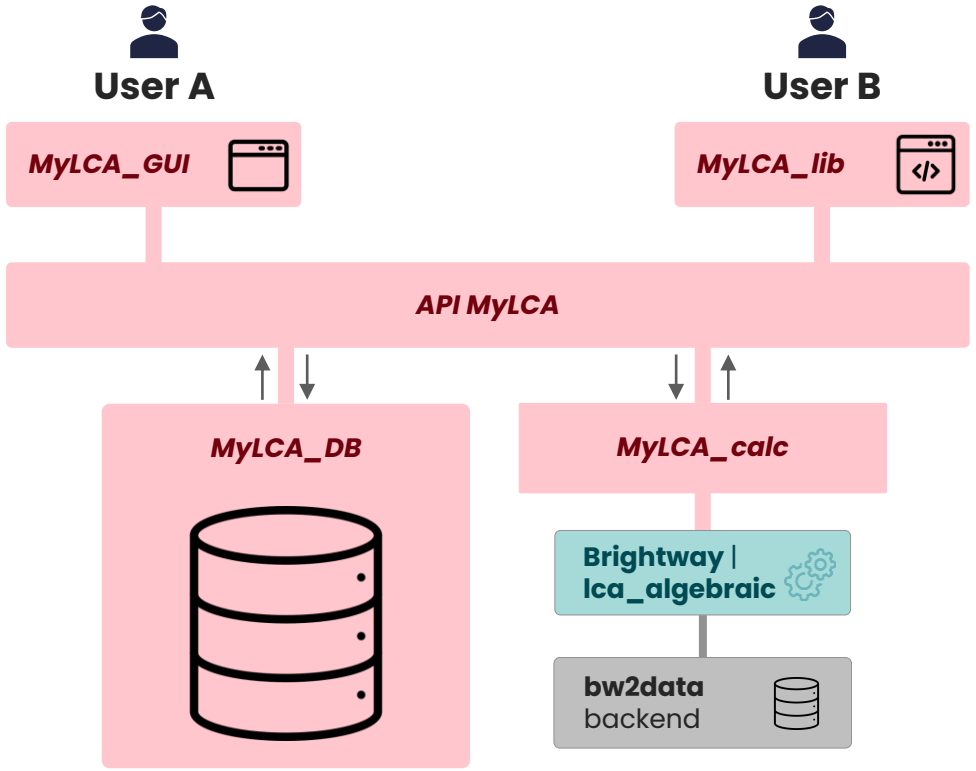
MBSE-based LCA web GUI

A shared, versioned, LCA models catalogue

The goal

MyLCA could provide it as a global solution with the following **benefits:**

- **Formalize concepts** to define **parametric**, **reusable**, and **sharable** LCA sub-models
- Serve them to both **Python** and **non-Python practitioners**.



⇒ An **internal catalogue** of **versioned parametric LCA models**, computable with Brightway, open to a **GUI** as much as to **Python** | Open source

Any questions?

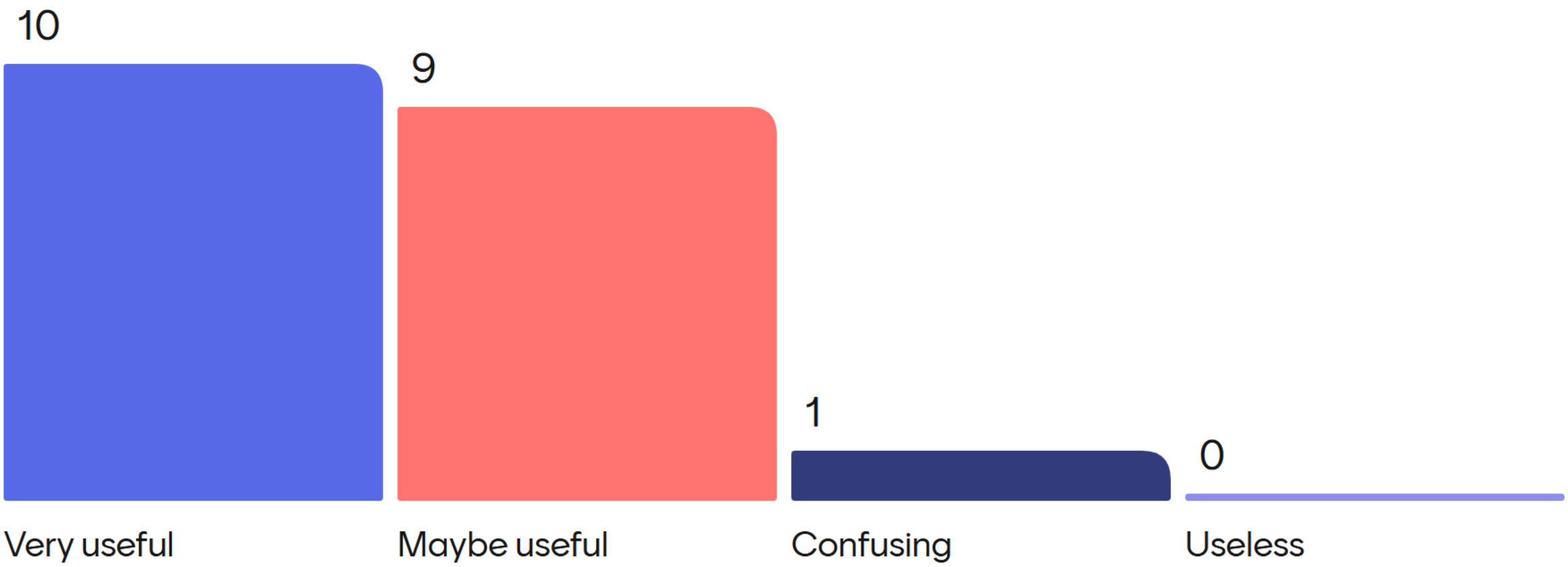
Ports · Containers · Templates · Instances · Switches · Parameters

3 Feedback gathering

Join at menti.com | use code 4597 2211

Mentimeter

What do you think of adding **ports** to activities and containers?



Show trends D

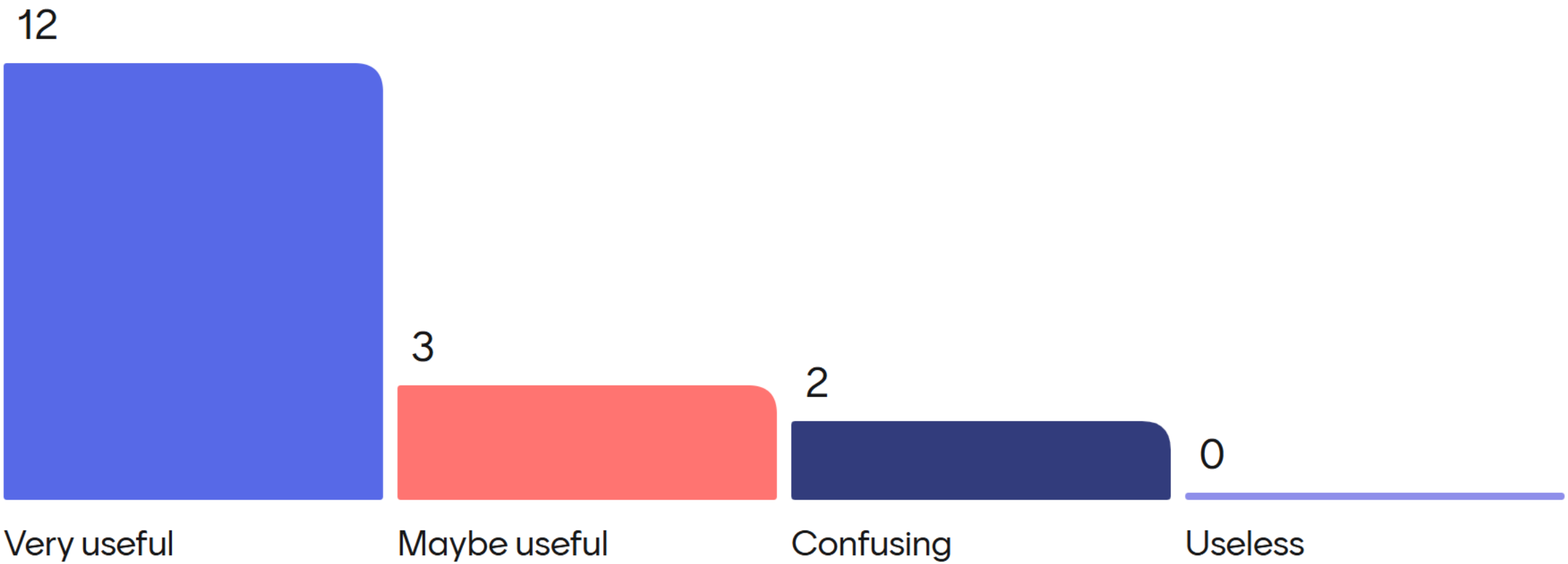
2

18 / 34

Join at menti.com | use code 4597 2211

Mentimeter

What do you think of the notions of **containers** | **templates** | **instances**?



Show trends D

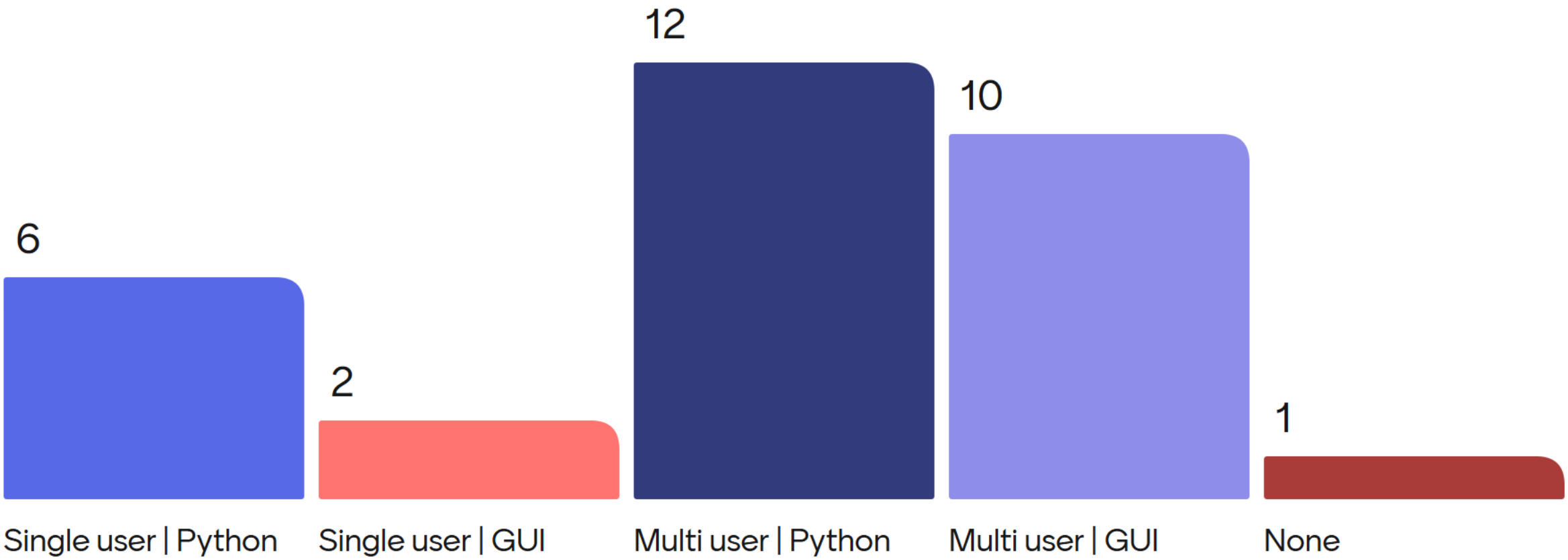
2

17 / 34

Join at menti.com | use code 4597 2211

Mentimeter

Which **MyLCA usage** would suit you the most?



2
17 / 34

Join at menti.com | use code 4597 2211

Mentimeter

Are there **things you could do with MyLCA** you can't do today?

i can but it more work?

Collaborate easily

No but definitely would be nice to habe a common interface for sharing

Build complex models more easily

Data models for processes

having fun with yet another silo :-)

Share with colleagues

Unable to answer

Maybe

Join at menti.com | use code 4597 2211

Mentimeter

Anything missing? Any other comments or remarks?

already thinking to make it usable within the sentier.dev infrastructure

Make sure that you require proper classifications for things entering in ports for future sharability

End presentation

2

2 / 34

Credits

Graphics

This icons on slide 2 are from **wichai.wi**, accessed on flaticon.com

Use of Generative Artificial Intelligence

9 successive prompts on the Claude Sonnet 5 LLM were used to help draft the storytelling of this presentation.
All the slides and graphics are then pure PowerPoint made entirely by the author.

Thank you for your participation!

Maneval, Victor – victor.maneval@cea.fr
Perdu, Fabien – fabien.perdu@cea.fr
Jamet, Didier
Goutelle, Mathieu
Maillot, Isabelle