Network LCA requirements in shipbuilding

2025-08-16

S. Majaniemi (VTT) and M. Eklund (Semantum)

Contents

1. Challenges in LCA data collection 2
2. Network LCA Service and Network LCA Data Exchange Portal 3
2.1 Network LCA Service 3
2.2 Network LCA Data Exchange Portal 5
3. Standardized data exchange formats 7
3.1 ILCD data format 8
3.2 Ecospold data format 8
4. Other requirements 9
5. References 10
Appendix 1: Standardized ICLD data exchange format 10
Appendix 2: API reference manual for VTT Network LCA Service 12
Appendix 3: Snapshots of Semantum's Network LCA Data Exchange Portal 17
A.1 Login 17
A.2 Add a new Company 18
A.3 Group Companies 19
A.4 Create a Template 20
A.5 Edit Template 21
A.6 Allow users to add dynamic entries 22
A.7 Sulca Conversion Template 24
A.8 Add new Parameters 24
A.9 Formula Builder 25
A.10 Create a Data Request 26
A.11 Fill in an Incoming Data Request – End Users 27
A.12 Generate a SULCA-Compatible Excel from a Completed Data Request 27

A.13 Export options.....29
A.14 Sulca Compatible Excel 30

1. Challenges in LCA data collection

As sustainability principles move from voluntary practice to regulatory requirement, the adoption of Life Cycle Assessment (LCA) and other environmental assessment methods is becoming increasingly vital for companies. The European Union has already introduced stringent regulations such as the Corporate Sustainability Reporting Directive (CSRD) and the EU Taxonomy, both of which demand transparent reporting and quantification of environmental impacts throughout entire value chains. The most standardized component of different assessment methods (environmental, social, governance) is LCA. As the ISO (14040)[1] standard, it consists of four phases. In phase one, the goal and the scope of the assessment are defined. In the second phase, an inventory analysis is performed followed by the life cycle impact assessment as phase three. Finally, as the last phase of the LCA, the results obtained will be interpreted according to the goal and scope. The inventory analysis includes the data collection phase, which is the most essential and time-consuming part in the LCA.

Collection of inventory data is seen laborious because the data are collected from various sources and because the data sources might be from a different organization or geographical location than the organization of the commissioner of the LCA study. This is especially true for shipbuilding, which relies on wide subcontractor networks.

Secondly, the data provider might regard some of their data confidential. This is because the recipes might be secret, i.e. the raw materials and the amounts of raw materials and/or the amount of energy consumed in the production process reveals the cost structure of the product.

Thirdly, the fact that the collection of accurate environmental data requires a lot of work, which cannot currently be automated effectively, does not encourage delivery of data without financial motivation, which could derive from regulative measures or e.g. carbon tax policies.

Finally, even if the companies in the value chain are willing to share data with each other, technological obstacles can hinder the process. In our previous report [add ref.], we have analyzed the current situation in the shipbuilding value chains and found several factors, such as incomplete or missing data exchange protocols, which slow down the flow of important information.

2. Network LCA Service and Network LCA Data Exchange Portal

Two independently operatable software pieces have been created during the project. It is possible to combine their functionalities in the future.

1. VTT has constructed the 2nd generation of its Network LCA **Service**, which is available to external users via VTT's collaborative simulation platform called Modelling Factory.
2. Semantum Oy has created software called Network LCA **Data Exchange Portal**, which focuses on the standardized data exchange of LCA data.

In a nutshell, Network LCA Service is a cloud service, which allows the users to conduct their own LCA analysis using a life-cycle assessment model, which can be opened to *e.g.* to all value chain member companies. No official policy has been established in the case of ship building sector as to the degree of openness level of the usage, several possibilities are possible. Network LCA Service does not currently support the use of data exchange formats, such as ILCD, which have been specifically designed for LCA data.

The Network LCA Data Exchange Portal, on the other hand, supports the exchange of LCA data in standardized ILCD/e-ILCD formats. Its core functionality is **not** the actual simulation of the LCA models but the **control of the data flows** in a structured manner according to ISO standards. The users of Data Exchange Portal can make formal data requests and answer requests from other users while keeping track of the message flows in an organized manner.

Both services mentioned above are cloud services, which require no installation or maintenance operations from their users, in principle. In practice, also local installations of these services is possible, in which case the maintainer of the service assumes responsibility of all aspects from software component updates to LCA model changes, data curation etc.

2.1 Network LCA Service

In the widest sense, the Network LCA Service is a tool which can be used for sustainability analysis and testing product design possibilities. In other words, with the aid of the Network LCA Service, the effect of different material compositions or production process choices on the LCA performance indicators could be studied. **Owing to the rapidity and analysis properties independent from other value chain/network members, the results can be utilized *e.g.* for product design in a manner not traditionally allowed by life-cycle assessments**. The tool also makes it possible to share the LCA results within the network of multiple operators or companies.

From the data flow point of view, the Network LCA Service idea is depicted in Fig. 1 below. Every partner of the network in the value chain (or value network) can edit and fill in their own

product/process data via a web user interface connected to the LCA software used in the computation of the results. The data format is **free** (**not ISO standard fixed**) and has to be separately agreed on with the maintainer of the shared LCA model (the Trusted operator, see Fig. 1 below). In the demos set up during the SusFlow project, the Trusted operator, VTT, has chosen the variables, which can be changed by the users of the demo models. Naturally, the current choices could easily be altered based on further negotiation.

Network LCA Service can be utilised differently depending on the availability of the LCA data and the data sharing policy of the network. Obviously, the biggest gains can be achieved if all the data is openly shared. However, as pointed out in Sect. *Challenges in data collection*, companies may not want, or are unable to share all the useful data. For such situations, the service can still be utilized for computation of results, whose sharing does not reveal business secrets openly. Typically, aggregate results, such as total network level carbon emissions is an example of a usefull sustainability indicator, which does not reveal individual contributions of companies in larger networks. Yet, being able to see one company's effect on the total footprint may help them to understand, how big (or small) role they play in making up the total footprint of the end product. Naturally, the Network LCA Service tool can also be used completely locally (for company specific products only) if the LCA model has been set up with sufficient resolution to allow this. This corresponds to the traditional way of utilizing life-cycle assessment to study sustainability restricted to single company's single product. While the current demonstrations in SusFlow project focus on the computation of the carbon footprint, obtaining a full-fledged LCA report with many more sustainability indicators is also possible.

To summarize, the Network LCA Service can be used in many different ways depending on the level of openness and the governance model of the network activities. Its near-term usage will most likely depend on what type of data the forthcoming legislation forces value network members to share with each other. If a member level LCA computation is allowed by the network policy, all (designated) members of the network members can also perform local LCA computations to study their own local footprints with the aid of the shared network level LCA model. In other words, the network members can independently of each other run tests to investigate the impacts of changes e.g. on material choices or manufacturing methods both locally and at the network level. This corresponds to the traditional use of LCA when network level results are excluded. It should be mentioned that the ability to see the network level results is also beneficial for policy planners who want to see the big systemic picture and formulate their policy setting action plans based on the observed and predicted data.

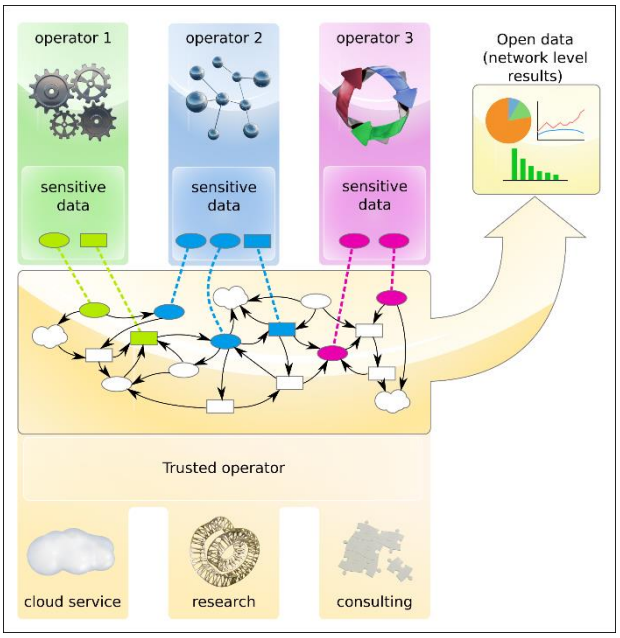


Figure 1: Building blocks of the Network LCA Service by VTT.

Fig. 1 presents the building blocks of the Network LCA Service from the modeling point of view. Various operators (network member companies) provide different data items into the collective, shared LCA model maintained by the Trusted operator, who is responsible for constructing the model in such a way that the important network level results do not reveal sensitive data items to network members. Naturally, the trusted operator, which could be an independent research institute (*e.g.* VTT) or the assembler of the final product, must be an entity who can be trusted to be impartial as regards handling of the data. In the extreme cases, the trusted operator's role could be restricted to only setting up the computational LCA model without access to the different operators raw data. Currently, VTT's Network LCA Service does not enable this type data masking, but technically this level of data protection is also possible.

2.2 Network LCA Data Exchange Portal

The two central ideals of the Network LCA Data Exchange Portal are both related to the organization of the LCA data exchange. As explained in Sect. *Challenges in data exchange*, the lack of universally accepted exchange standards hinders the utilization of LCA data, even if it would be usable for other reasons. Semantum's Network LCA Data Exchange Portal answers this challenge. It formalizes the message passing related to the exchange of sustainability information. In addition, it leverages on ISO standards in the exchange of LCA data. The most relevant data formats utilized for sustainability data are ILCD and eILCD. Since these formats are not familiar to many of the smaller companies only getting familiar with the LCA requirements themselves, Semantum's Data Exchange Portal also supports data exchange in MS Excel format. It should be stressed that Microsoft and other commercial provider associated data formats are not universal standards enjoying the status and applicability conditions of global standards such as those of ISO. The

former are subject to the whims of their issuer companies. However, given the wide spread use of Excel, it makes sense to also provide some functionality to facilitate its use in LCA data exchange.

It should be remembered that also the VTT's Network LCA Service currently operates based on data exchange format, which is not a universal standard. Therefore, the combination of the data exchange format based on ISO standards (Semantum) with the shared computational model for all network members as described in Sect. *Network LCA Service* (VTT) would the most influental solution to future development of the networked impact assessments.

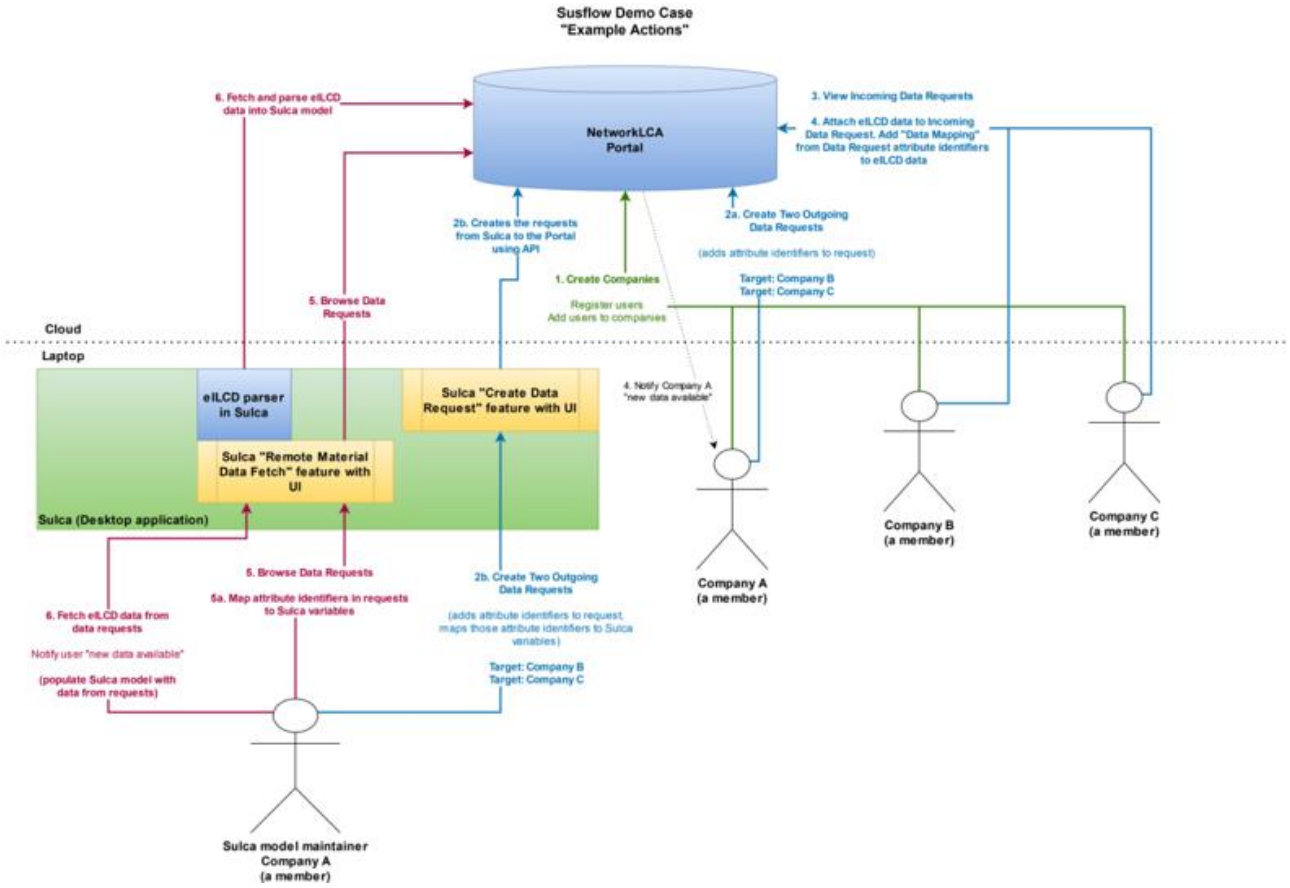


Figure 2: Building blocks of the Network LCA Data Exchange Portal.

The main components of Semantum's Network LCA Data Exchange Portal (NetworkLCA Portal for short) can be seen in Fig. 2 above. Companies A, B and C correspond to the various "Operators" designated in Fig. 1. The references to "Sulca" refer to the LCA software utlized currently by VTT and Semantum to conduct life-cycle assessments. This computational tool could be changed in the future according to the preferences of the companies, but at the moment it is the choice utilized for the consortium leverl demos of SusFlow project.

3. Standardized data exchange formats

To exchange Life Cycle Assessment (LCA) and Life Cycle Inventory (LCI) data, a standardized format must be selected. There are several life cycle data exchange formats, and here we present the most widely used.

Emerging LCA databases might start out with only one format but should ideally strive to support several different possibilities. The technical ISO standard (ISO/TS 14048:2002) provides guidance to life cycle data developers, as it sets the requirements and a structure for a data documentation format, to be used for transparent and unambiguous documentation and exchange of LCA and LCI data.

The ILCD and ecoSpold (v1 and v2) formats represent the two most well-established data exchange formats currently available. Both ILCD and ecoSpold are compliant with ISO/TS 14048 and are based on Extensible Markup Language (XML). Conversion between these common data exchange formats is supported by available converters, e.g. in LCA software or the converter function in the GLAD network.

The development of the International Reference Life Cycle Data System (ILCD) was driven by the need for: (i) a data format for the European Reference Life Cycle Database (ELCD), (ii) a common format to support data exchange (import and export) of the ELCD reference datasets with other databases and software tools, (iii) a common format to be used to exchange LCA datasets among all relevant LCA tools and databases (e.g. for LCA information transfer along supply chains) and for data networks, and (more recently) and (iv) the development of data under the Environmental Footprint scheme (PEF/OEF). The ILCD format is supported by major LCA software applications.

The ecoSpold format is the result of several iterative evolutions of XML-based data formats for LCA data, beginning in the 1990s. While sometimes associated primarily with the EcoInvent LCI database, it is an open-source format, and its most recent version is the ecoSpold v2 format (Meinshausen et al. 2016). The ecoSpold format is supported by most major LCA software applications.

JSON-LD is a technical alternative to XML that combines ILCD and EcoSpold. JSON-LD is a format based on JavaScript Object Notation for Linked Data (JSON-LD) developed by GreenDelta. Besides the aim to reduce the effort for implementation and remove inconsistencies between the ILCD and ecoSpold formats, there are other advantages, such as being human-readable and the ease of integration into web-applications. JSON-LD was implemented as one of the formats used by openLCA in 2015, but use of this format in other LCA software is still limited.

The Product Environmental Footprint / Organizational Environmental Footprint (PEF/OEF) was developed in ILCD format with significant changes to the elementary flow nomenclature system. These changes must be considered during the development of national databases due to the need for homogeneity in nomenclature among the datasets to be created using the ILCD system.

3.1 ILCD data format

In order to get an idea, how does standardized data exchange formats such as ILCD look like, the reader should consult App. 1. In order to convert LCA data into ILCD/eILCD format, the use of specific software products is required, which may increase the willingness of the value chain companies to use the more universal exchange formats. For example, presently we use an open source software tool called OpenLCA to perform the conversion of data.

The ILCD data format is an XML-based format with eight available dataset types for the elements in the database 12. The following are the dataset types of the ILCD format:

1. **Process**: This includes the input and output flows for a system or unit process.
2. **Flow**: This includes the flow properties and categories for a given flow.
3. **FlowProperty**: This includes the properties of a given flow.
4. **LCIAMethod**: This includes the methods used to calculate the impact assessment results.
5. **UnitGroup**: This includes the units of measurement used in the database.
6. **Source**: This includes the sources of the data used in the database.
7. **Contact**: This includes the contact information for the data provider.
8. **LifeCycleModel**: This includes the entire models consisting of ILCD process data sets 1.

XML is also part of the data representation of the LCA computational models, which are mapped into a co-simulation standard known as Functional Mock-up Interface (FMI) standard. This standard is not a global standard such as ISO, but it is utilized by several major players in the simulation software business and one of the most likely candidates at the moment for a truly universal simulator data exchange. FMI standard is currently utilized by VTT based on our best guess for the next dynamic scientific simulation software data exchange format, which ensures interoperability of the software from different vendors. In FMI format, XML representation takes care of the cataloging of the input and output variables needed by different simulation programs to exchange data. Currently, however, VTT's Network LCA Service only utilizes this standard for simulation data exchanges, not for LCA data exchanges in general. To get the best of both worlds, the functionality of the VTT and Semantum software products should be combined.

3.2 Ecospold data format

XML is a standardized data representation format, which is utilized by various types of software, which are not related to sustainability data as such. However, due to their utility, many companies associated with LCA make use of it. For example, the EcoSpold format is an XML-based format that is widely used in the LCA community [3]. It is a flexible and extensible format that can be adapted to meet the needs of different users and applications. EcoSpold is used by many LCA software

tools, including openLCA, SimaPro, and GaBi. The format is designed to be compatible with other data formats, which makes it easier to integrate with other systems. EcoSpold is used to store LCA data, including inventory data, impact assessment methods, and characterization factors.

The EcoSpold and ILCD formats are both XML-based formats used in the LCA community [3]. EcoSpold is an older format that was developed in the early 2000s, while ILCD is a newer format that was developed by the European Commission in 2010. Both formats are designed to facilitate the exchange of environmental information and create a common basis for consistent, robust and quality-assured life cycle data.

While both formats are similar in many ways, there are some key differences between them. One of the main differences is that EcoSpold is a more simplified format than ILCD, which makes it easier to use for smaller datasets [4]. ILCD, on the other hand, is a more complex format that is better suited for larger datasets. Another difference is that ILCD is more flexible and extensible than EcoSpold, which means that it can be adapted to meet the needs of different users and applications. ILCD is also more interoperable with other data formats, which makes it easier to integrate with other systems.

Many tools exist to convert data to ILCD format [1]. One of them is the openLCA format converter. This tool can convert various LCA data formats from one format into the other, including EcoSpold01 to ILCD, EcoSpold02 to ILCD, ILCD to EcoSpold01, and ILCD to EcoSpold02. Converted files can also be used by other LCA software packages, beyond openLCA. The converter runs as a stand-alone application on Windows, Mac, and Linux operating systems.

Another way to convert your own databases to and from ILCD is to use the export function from SimaPro [2]. This function converts datasets into ILCD using a basic mapping. Afterwards, you can add the entry-level documentation in an XML editor. If you would like some help or have a specific ILCD database to convert – to and from SimaPro – you can get in touch with the SimaPro support team.

4. Other requirements

The additional requirements presented in this section are only necessary when utilizing the VTT collaborative simulation platform (Modelling Factory) and its API service as a hosting environment for the LCA models given its current setup.

All simulation models (LCA models belong to this category even though they are not dynamic simulation models) must comply with Functional Mock-Up Interface (FMI) version 1.0 or 2.0, Co-

Simulation. Since the VTT implementation runs on top of a Linux cluster, the FMI model uploaded into our platform, must also contain the binary files necessary for Linux environment.

VTT's API service has been implemented as Docker imager in Kubernetes environment. It is possible to use alternative setups depending on the provider of the Network LCA Service. As stated previously, VTT is currently acting as the Trusted operator during the SusFlow project, but in the future, other parties may assume responsibility for this role, as well.

Both the services (VTT and Semantum contributions) can be run as a cloud service but separate agreements must be made the continued use, if cloud access is the preferred mode. Alternatively, local installations can also be negotiated.

5. References

[1] https://www.openlca.org/format-converter/

[2] https://pre-sustainability.com/articles/the-ilcd-format-solving-lca-data-exchange-problems /

[3] https://helpdesk.lifecycleinitiative.org/distribution/data-formats/

[4] https://link.springer.com/article/10.1007/s11367-020-01748-2

Appendix 1: Standardized ICLD data exchange format

Here is a partial example of an **XML format** file for an ILCD dataset 1 (European Platform on LCA | EPLCA (europa.eu)):

```
<?xml version="1.0" encoding="UTF-8"?>
<ProcessDataSet xmlns="http://www.EcoInvent.org/EcoSpold02" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.EcoInvent.org/EcoSpold02 EcoSpold02.xsd">
  <processInformation>
    <dataSetInformation>
      <dataSetID>ILCD-123456</dataSetID>
      <dataSetTitle>Production of 1 kg of polyethylene terephthalate (PET) bottles</dataSetTitle>
      <dataSetVersion>1.0</dataSetVersion>
      <dataSetAuthor>John Doe</dataSetAuthor>
      <dataSetLanguage>en</dataSetLanguage>
      <dataSetLicense>CC BY-SA 4.0</dataSetLicense>
      <dataSetPublicationDate>2022-01-01</dataSetPublicationDate>
```

```
      <dataSetModificationDate>2022-01-01</dataSetModificationDate>
      <dataSetComment>This dataset was created for the purpose of demonstrating the ILCD data
format.</dataSetComment>
    </dataSetInformation>
    <processInformation>
      <processID>ILCD-123456-1</processID>
      <processType>unit process</processType>
      <processName>PET production</processName>
      <processDescription>Production of 1 kg of polyethylene terephthalate (PET) bottles</processDescription>
      <geography>Europe</geography>
      <technology>Conventional</technology>
      <referenceYear>2020</referenceYear>
      <dataSetAuthor>John Doe</dataSetAuthor>
      <dataSetLanguage>en</dataSetLanguage>
      <dataSetLicense>CC BY-SA 4.0</dataSetLicense>
      <dataSetPublicationDate>2022-01-01</dataSetPublicationDate>
      <dataSetModificationDate>2022-01-01</dataSetModificationDate>
      <dataSetComment>This dataset was created for the purpose of demonstrating the ILCD data
format.</dataSetComment>
    </processInformation>
    <flowData>
      <flow>
        <flowID>ILCD-123456-1-PET</flowID>
        <flowName>PET</flowName>
        <flowCategory>Product</flowCategory>
        <flowSubCategory>Polymer</flowSubCategory>
        <flowUnit>kilogram</flowUnit>
        <flowComment>Polyethylene terephthalate (PET) is a thermoplastic polymer resin of the polyester
family.</flowComment>
      </flow>
      <flow>
        <flowID>ILCD-123456-1-Energy, electricity, high voltage</flowID>
        <flowName>Energy, electricity, high voltage</flowName>
        <flowCategory>Electricity</flowCategory>
        <flowSubCategory>High voltage</flowSubCategory>
        <flowUnit>kilowatt hour</flowUnit>
        <flowComment>Electricity generated from high voltage sources.</flowComment>
      </flow>
      <flow>
        <flowID>ILCD-123456-1-Energy, electricity, medium voltage</flowID>
        <flowName>Energy, electricity, medium voltage</flowName>
        <flowCategory>Electricity</flowCategory>
        <flowSubCategory>Medium voltage</flowSubCategory>
        <flowUnit>kilowatt hour</flowUnit>
        <flowComment>Electricity generated from medium voltage sources.</flowComment>
      </flow>
      <flow>
        <flowID>ILCD-123456-1-Energy, electricity, low voltage</flowID>
        <flowName>Energy, electricity, low voltage</flowName>
        <flowCategory>Electricity</flowCategory>
```

Appendix 2: API reference manual for VTT Network LCA Service

The functionalities of the REST API utilized for LCA (and other) modelling are presented below in Table 1. Sulca LCA models are cast into the Functional Mock-up Interface standard format and they can be accessed via the application programming interface offered by the VTT service. The service is equipped with graphical user interface but it can be fully used via command line tools entirely, if necessary.

Feature	Explanation
Access	
CI Work room access	All API calls target a single workroom or a single numerical simulation experiment (= "experiment", for short) inside a workroom, unless otherwise stated. The chosen workroom is provided in the HTTP request as a field. In case the field is not specified, API call will be directed to the user's private workroom. All API calls are also authenticated, meaning, if a user does not have access to a workroom, she cannot perform any API actions on that workroom. A workroom admin is a user who has created a workroom or has been promoted to workroom admin by another workroom admin. Being the workroom admin of one workroom does not automatically give access to all other workrooms. Rather, the admin permissions are limited to specific workrooms. Workroom admins can perform all actions that affect the workroom of which they are admin.
CI web UI: Https	Access protocol (browser access)
CI API: Https	Access protocol (external/third party programmatic access)
Explicit call commands via CI API (Kubernetes cluster)	
UploadFMU	Upload a new FMU model to the server to a single workroom, available for all users with access to the chosen workroom. This call auto-generates a unique identifier "modelId" that can be used in other API calls. Alternatively, a chosen "modelName" can be used to reference the model instead.
DownloadFMU	Download an FMU model from the server to your local computer. Only models for the original uploader, or the workroom admin, has set "download enabled" option to be true, can be downloaded using this API.
DeleteFMU	Delete an FMU model by its modelId or modelName, from the server. Experiments and results that used the model will still be browsable by users.
InspectFMU	Inspect an FMU model contents on the server, using the modelId or modelName. Returns a list of all FMI input/output variables listed in the model, and the details of each variable.
ListFMUModels	List all available FMU models in a specified workroom, returning a JSON array with the information of all the models, showing their modelId and modelName.
ListResultFiles	List all available result files of a single completed experiment (see API calls CreatePipeline and fmi2Instantiate below). A single experiment refers to a single FMU model's simulation results.

DeleteResult	Delete a single completed experiment and all its result data. Users are only allowed to delete results they have created themselves, unless the completed experiment exists in a workroom where the user is a workroom admin.
DownloadResultFile	Download a single result file from a single completed experiment, using the result file's name.
DownloadReport	Download PDF report of a completed experiment. PDF shows all subscribed FMU output variables in plots, one variable per plot, two plots per A4 page. Naturally, in order for the plot to be generated properly, the FMU must provide an output, which can be visualized with the CI visualization mechanisms. Currently, only time-series plots are supported (see Sect. **Visualization functionalities** of this Table for further options). Future versions of CISS are planned to support other data sources than FMU output variables, such as plots of csv-files, tables and images. In such a case, an HTML file will be generated instead of a PDF.
CreatePipeline	Creates a new pipeline of FMU experiments. A pipeline is a "sequence" of FMU experiments, where the outputs of a previous experiment can be used as inputs for the next experiment. This API call can be used to create a pipeline *template* under a specific workroom, that others can then use to create actual pipelines with actual experiments. In case the user only wishes to run a single experiment without passing the created result files onwards for other FMUs to process further, the experiment is called a *singleton* experiment. During experiment creation (regardless if it was an experiment in a pipeline or a singleton experiment from fmi2Instantiate call), the caller must specify, which output variables they are interested in monitoring for time-series data, i.e. which outputs the experiment should subscribe to.
DeletePipeline	Delete a pipeline (and all its experiments and results) or a pipeline template. User can only delete pipeline he/she has created, or pipelines that exist in workrooms where the user is a workroom admin.
ListPipelines	List all available actual pipelines under a specific workroom.
ListTemplates	List all available pipeline templates under a specific workroom.
ListExperiments	List all available results under a specific workroom. Contains all the numerical experiments created in the specified workroom.
FMI Life-cycle functions via API	Low-level function calls for controlling an individual FMU experiment. Either use pipelines (see **CreatePipeline**) or create an experiment using **fmi2Instantiate** API call.
FMI call order	The call order of the FMI API calls is as follows: 1. fmi2Instantiate 2. (Optionally) fmi2UploadCustomResourceFile and any amount of fmi2SetString, fmi2SetReal, fmi2SetInteger and fmi2SetBoolean calls for -parameter- type variables. 3. fmi2ExitInitializationMode 4. Any amount of fmi2GetXXX and fmi2SetXXX functions and fmi2DoStep (where XXX is String, Real, Integer, Boolean) 5. fmi2Terminate
fmi2Instantiate	Create a new numerical experiment from the chosen FMU model, using the modelId or modelName of the FMU model. No simulations can be performed without such experiments. Pipelines use multiple experiments in a row, see CreatePipeline, while fmi2Instantiate creates a single experiment. During experiment creation (regardless if it was an experiment in a pipeline or a singleton experiment from fmi2Instantiate call), the caller must specify

	which output variables they are interested in monitoring for time-series data, i.e. which outputs the experiment should subscribe to.
fmi2UploadCustomResourceFile	Upload any binary file to the FMU experiment's working directory. These files can be e.g. input parameter files, or data files that the model needs to be performed. The files will be uploaded into the internal DB of Collaboration Interface, not ANSYS-Granta DB if not specifically required by the uploader.
fmi2ExitInitializationMode	After calling fmi2Instantiate, a user can call "fmi2SetString", "fmi2SetInteger", "fmi2SetReal" and "fmi2SetBoolean" for parameter variables (see below). They can also call "fmi2UploadCustomResourceFile". Once custom files and parameters have been set, the FMI experiment must be marked as no longer being in an "initialized" state. Calling fmi2ExitInitializationMode is mandatory before calling any other FMI experiment related API calls.
fmi2Terminate	Cleanup and shutdown a single experiment, marking it as completed. The experiment's results can now be viewed, once uploaded to the result server (this operation can take a while if the result files are large). To delete the experiment and its results, use DeleteResult API call instead. If you wish to remove the results produced by an experiment, you must call fmi2Terminate, then call DeleteResult.
API commands for controlling FMUs (getters, setters, stepper)	Run-time control of FMUs.
fmi2GetReal	Given an FMU experiment, get the current value for any “real” (i.e. double) type output variables from the FMU model experiment.
fmi2GetInteger	Given an FMU experiment, get the current value for any “integer” type output variables from the FMU model experiment.
fmi2GetBoolean	Given an FMU experiment, get the current value for any “boolean” (i.e. “true” or “false” or “1” or “0”) type output variables from the FMU model experiment.
fmi2GetString	Given an FMU experiment, get the current value for any “string” type output variables from the FMU model experiment.
fmi2SetReal	Given an FMU experiment, set parameter or input variables (real, i.e. double type) for the FMU model, which it will then use the next time fmi2DoStep is called (see below). Parameters can only be set before "fmi2ExitInitializationMode" has been called.
fmi2SetInteger	Given an FMU experiment, set parameter or input variables (integers) for the FMU model, which it will then use the next time fmi2DoStep is called. Parameters can only set before "fmi2ExitInitializationMode" has been called.
fmi2SetBoolean	Given an FMU experiment, set parameter or input variables (Booleans) for the FMU model, which it will then use the next time fmi2DoStep is called. Parameters can only set before "fmi2ExitInitializationMode" has been called.
fmi2SetString	Given an FMU experiment, set parameter or input variables (strings) for the FMU model, which it will then use the next time fmi2DoStep is called. Parameters can only set before "fmi2ExitInitializationMode" has been called.
fmi2DoStep	Given an FMU experiment, step the FMU model forward X seconds. The FMU model uses inputs and parameters and calculates new outputs. For FMU models without a concept of “time”, fmi2DoStep is used to trigger a computation based on provided inputs and parameters. It does not matter, which value has been assigned to the step interval in this case (via the API call or via the UI). In this case the FMU will execute only once and output the results.

API commands for reading time-series result data from FMU experiments' output variables	
ListTimeSeriesVariables	List the available output variables that were stored as time-series results during an FMU experiment. Only subscribed values during the simulation were stored and will listed by this API call, and only after fmi2Terminate has been called for the experiment (or the experiment is terminated by the pipeline it is running in).
ReadTimeSeries	Get the time-series values of the variables that were stored as output from an FMU experiment. Only subscribed values during the simulation were stored and will listed by this API call, and only after fmi2Terminate has been called for the experiment (or the experiment is terminated by the pipeline it is running in).
User guided operations via CI web UI	All the same functionalities are available as for the API detailed above. Additional functionalities, which are set explicitly through UI controls by the user are detailed below. The functionalities described below are not literal API commands, rather descriptions of what a user can do via the web UI.
Initialize simulation	This is the same action as calling "fmi2Instantiate" for an FMU model. In the UI, the chosen FMU model (inside the chosen workroom) will be used to create a new experiment. In the UI, the parameters will be asked for before the experiment is started, but the provided parameter values are still sent under-the-hood by the server to the created experiment after the experiment starts, thus respecting the order of calls as specified in the fmi2Instantiate API call.
Set simulation parameters	Provide necessary input parameters to guide the numerical solver. This is the same as calling fmi2SetXXX (where XXX is String, Real, Boolean or String) before fmi2ExitInitializationMode has been called.
Upload custom resources	Before starting the experiment properly, the user can upload custom resource files to the working directory of the experiment. This is the same as calling fmi2UploadCustomResourceFile before fmi2ExitInitializationMode has been called.
Step forward	Simulate an experiment forward in time for FMU's that have a concept of time. For a "batch simulation" type models, will execute a single calculation loop. Usually exactly 1 step is needed for such batch simulations. This is the same as calling fmi2DoStep API call.
Get	UI will monitor the output variables that were subscribed to and show them in the UI. Note that only FMI model output variables are monitored and converted automatically into a time-series plot at the end of the experiment. This is the same as calling fmi2GetXXX (where XXX is String, Real, Boolean or String) for all variables that were subscribed to.
Finish experiment / Discard experiment	The user has the choice to finish an experiment at any time, marking it as completed and allowing the CI server to store the result files. The user can alternatively cancel or discard the experiment at any time, removing the experiment from the database and removing all its result files.
Set up/delete a work room	User can set up a new workroom and delete a workroom he/she is a workroom admin of. By default, the newly created workrooms are always "private" (since the user that created them is the only one that has access to it at the time of creation). The user can invite other users to the workroom at any time and promote them to work room admin as needed.
Browse work rooms & pipelines & templates & models	User can browse workrooms, pipelines, pipeline templates and simulation models he/she has access to. This is usually done on a per-workroom basis, i.e. the user chooses a specific workroom and can see all items available in that workroom. Items in the public workroom (the work room members of which all users are) are always visible in all workrooms.

Browse model input & output variables	Within a workroom, the user can see the input/output variable listing of a model published in the same work room. This listing enables him/her to use the variable names in the pipelining of the model under inspection with other models the user has access rights of. This is the same as the InspectFMU API call.
Filter	When browsing, different types of filters can be applied on UI when searching for suitable work rooms/pipelines/models etc.
Perform a stand-alone numerical experiment	Execute a single stand-alone model FMU. This option takes the user through a series of UI pages where parameters, custom resources, stepping, setting values and getting values is performed. See "Initialize simulation" (at the beginning of this section) for more information.
Perform a pipelined numerical experiment ("chained run")	Execute several chained FMU model experiments in series. The outputs of a previous experiment can be used as the inputs and parameters for the next experiment. This is the same as CreatePipeline API call.
Generate PDF report	Download a PDF report of a completed FMU experiment. This is the same as DownloadReport API call.
Public work room functionalities	
Public work room contents	All authenticated users of Collaboration Interface have access to a fully public workroom, which serves all registered users. When entering CI UI, the user is placed in his/her personal (private) workroom. In addition, he/she will be made automatically a member of the CI public workroom, which serves all the registered CI users. In other words, all the models, pipelines and experiments published in the public workroom are available to all registered users.
Database access	
Collaboration Interface internal database for Work Rooms (Simantics DB)	FMU storage is taken care of by Simantics database via which FMU experiments to be run can be instantiated. Workrooms, registered users and their permissions, FMU models, pipelines, experiment definitions, are all stored in the Simantics database.
Collaboration Interface internal database (binary-data experiment result database, capability to show time-series data for FMU output variables, separate from Simantics *semantic* DB)	Final results produced by the FMU experiments are stored in a separate working area managed by another database (not Simantics triple store). All numerical experiments will store their results (for example time-series output) in this database. Results in this database are accessible by the CI server and result files and time-series data can be shown to authorized users. Direct access is not possible, as all requests must go through the CI HTTP UI or API, which checks if a user has permissions to read the results. Intermediary results for experiments are only stored in worker instance's local file system and will only be sent to the binary-data storage for completed experiments.
External (Granta) database	Communication via Granta SDK. Tested and verified.
Authentication/authorization	
Access authorization	Token based. Tokens provided by the KeyCloak implementation by the system admin. Also, third party access control can be arranged.
DB write results	UUID-based results write. Applies to all 3 different databases mentioned above.
Visualization functionalities	
Currently implemented visualization functionalities within Collaboration Interface	Currently, the Collaboration Interface visualization capabilities are limited to simple 2D plots (= time-series plots) where the output variable (free of choice) changes as a function of the simulation time steps (successive events determined by the FMI simulation protocol). Any result files produced in the numerical experiments can also be downloaded by the user for further processing with his/her own visualization software tools.
Collaboration Interface independent visualization functionalities	**Owing to the API of Collaboration Interface, all users of CI can implement their own UIs with their preferred visualization properties, which can be**

	implement with their preferred techniques and simply use the simulation model FMUs hosted by CI as computational engine that provides the data to be visualized. These "external" UIs are not hosted by CI or stored in the internal databases of CI. They are maintained and hosted by the users on the users' own servers.
Possible extension: Modelling Factory type work rooms with Simupedia visualization functionalities	Simupedia-based work room visualization capabilities can be brought about for Collaboration Interface users. The drawback is that in pipelined simulations, where outputs of many models should be combined into a single UI, more powerful visual representations could only be achieved by programmatic methods which require training of the UI builders (Simupedia widgets do support the use of many standard methods such as embedding Javascript or some other front-end technologies) but there is usually a need to 'write some glue-code', which enables efficient communication between different numerical solvers (possible packaged into their own FMUs) etc.

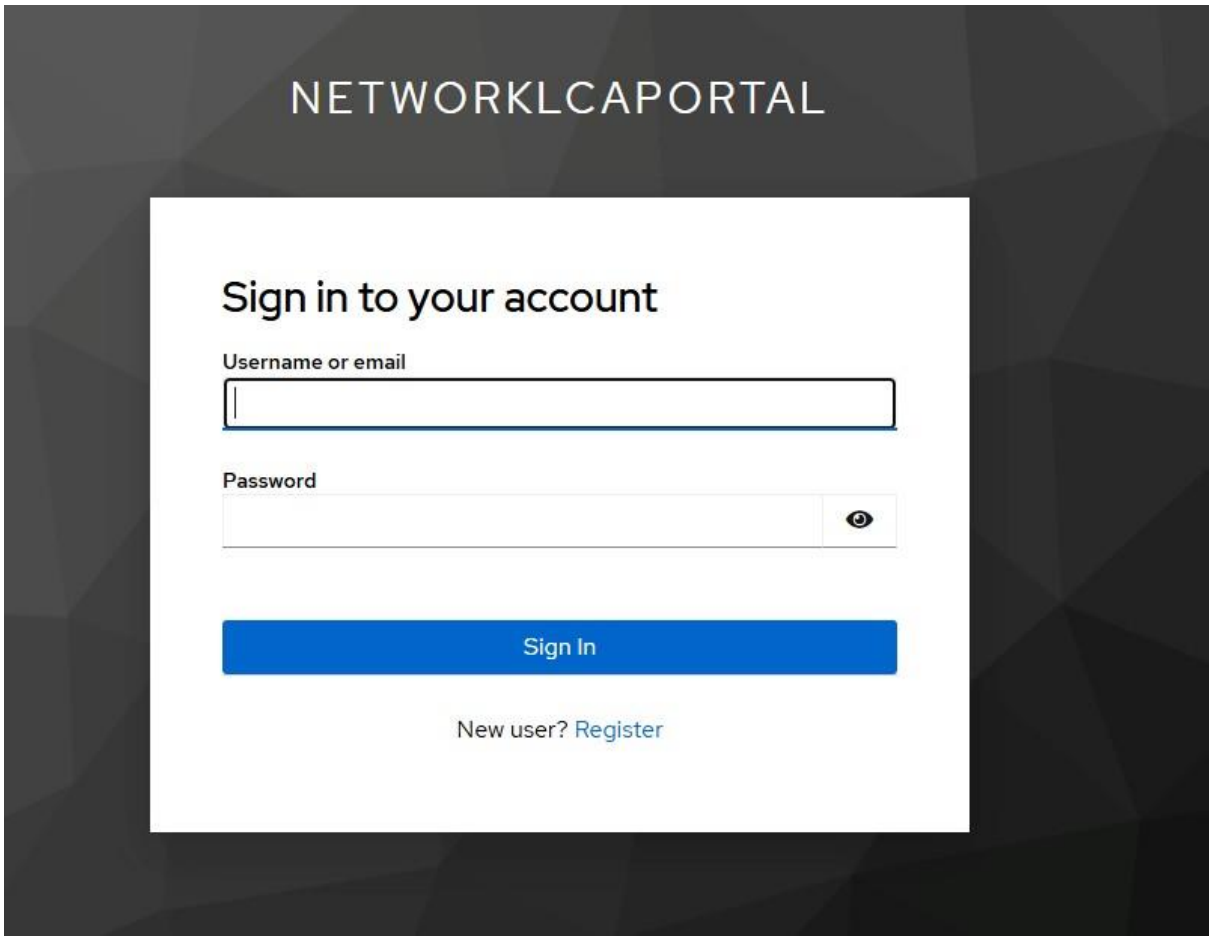
Table 1: Summary of the functionalities of the Modelling Factory API

Appendix 3: Snapshots of Semantum's Network LCA Data Exchange Portal

Below, you can find various snapshots of Semantum's Network LCA Data Exchange Portal functionalities from Login to various ways of organizing the data exchange between network participants.

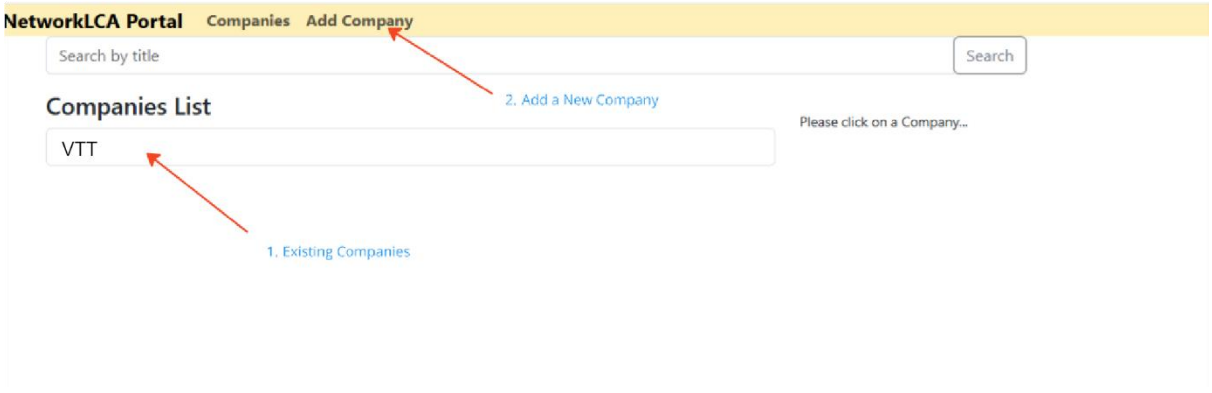
A.1 Login

Use your username and password to log in. If it's your first time logging in, use the temporary password provided via email and update your password.

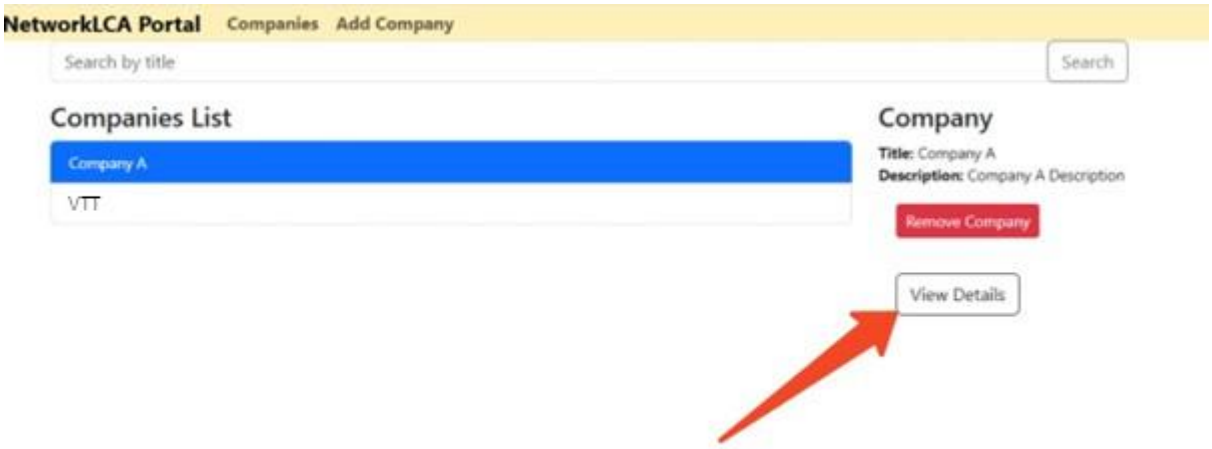


A.2 Add a new Company

1. Add Company → Provide company name and Description → Create Company



2. View Company details

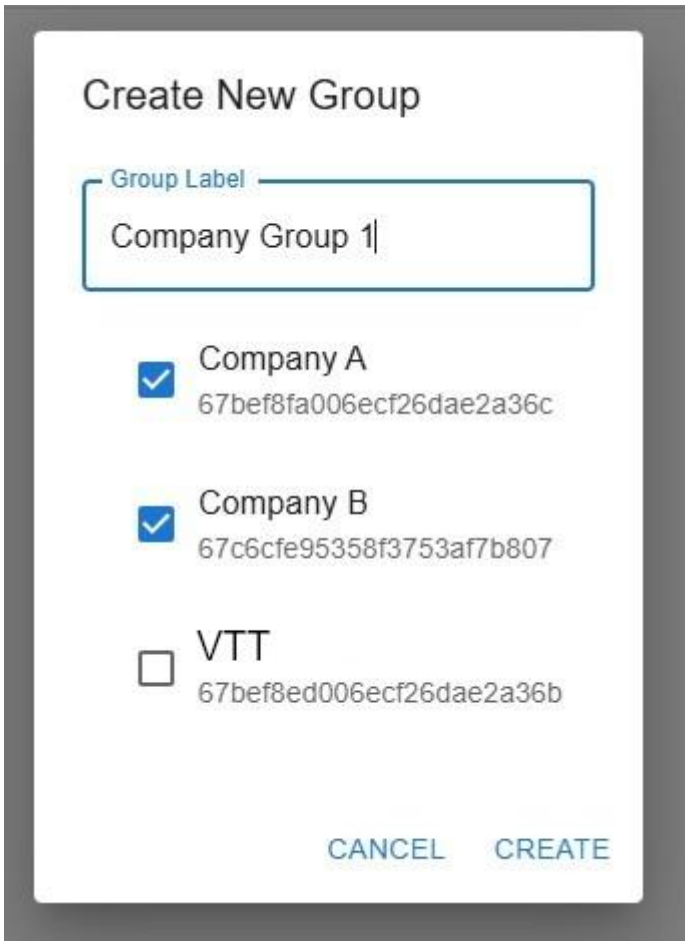


A.3 Group Companies

Users can group companies, allowing them to select a group and send a data request to all participants in that group at once.

1. Select the company and then View Company Details

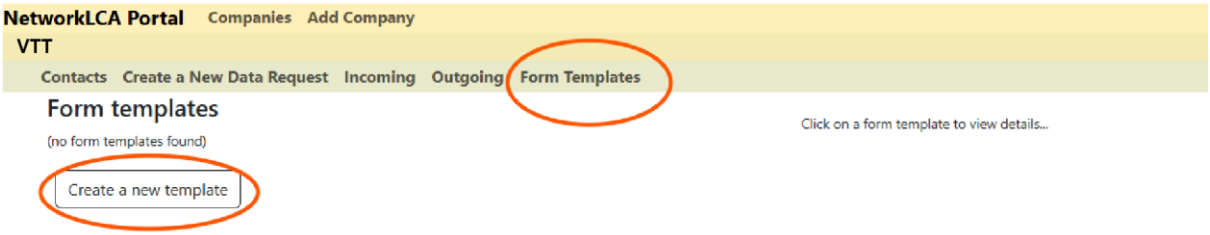
2. Contacts → Add a Contact Group
3. Create the group by selecting Companies



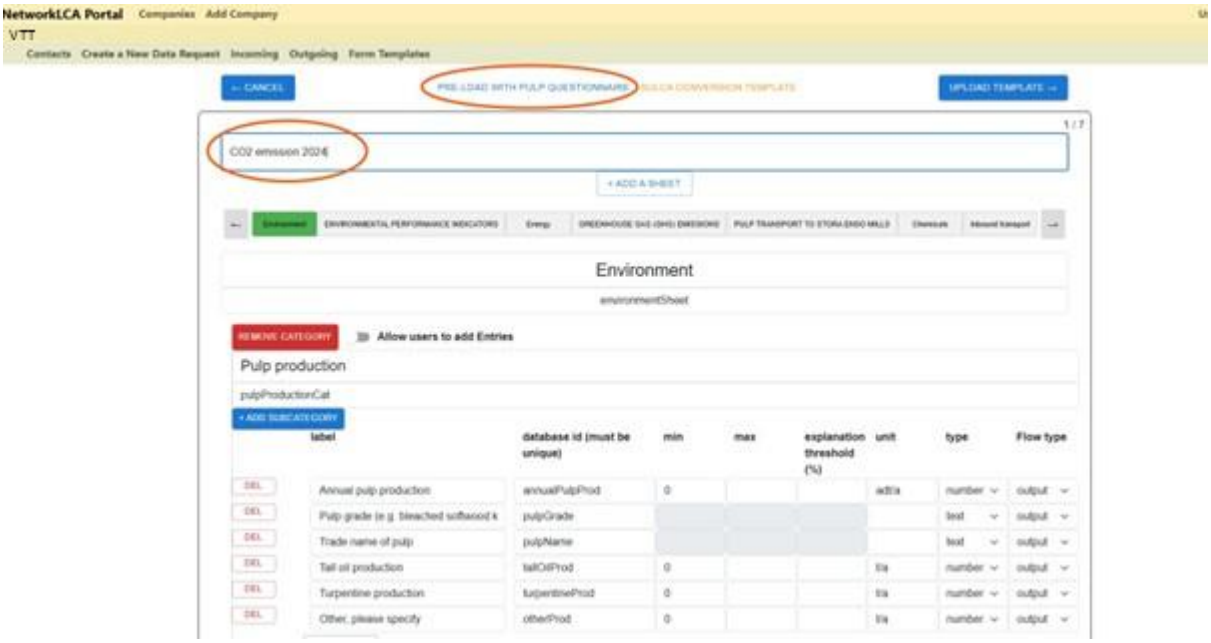
A.4 Create a Template

Users can create new templates from scratch or use a preloaded existing Pulp questionnaire. The created form templates can be stored for future use.

1. Form Templates → Create a new template



2. Load pulp questionnaire by pressing PRELOAD WITH PULP QUESTIONNAIRE button 3. Provide appropriate label for the template



A.5 Edit Template

The following options allow users to edit templates.

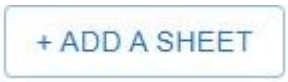
REMOVE CATEGORY - Remove Existing categories

+ ADD CATEGORY - Add new categories

+ ADD SUBCATEGORY - Add new Subcategories

- Add new fields

- Delete existing fields



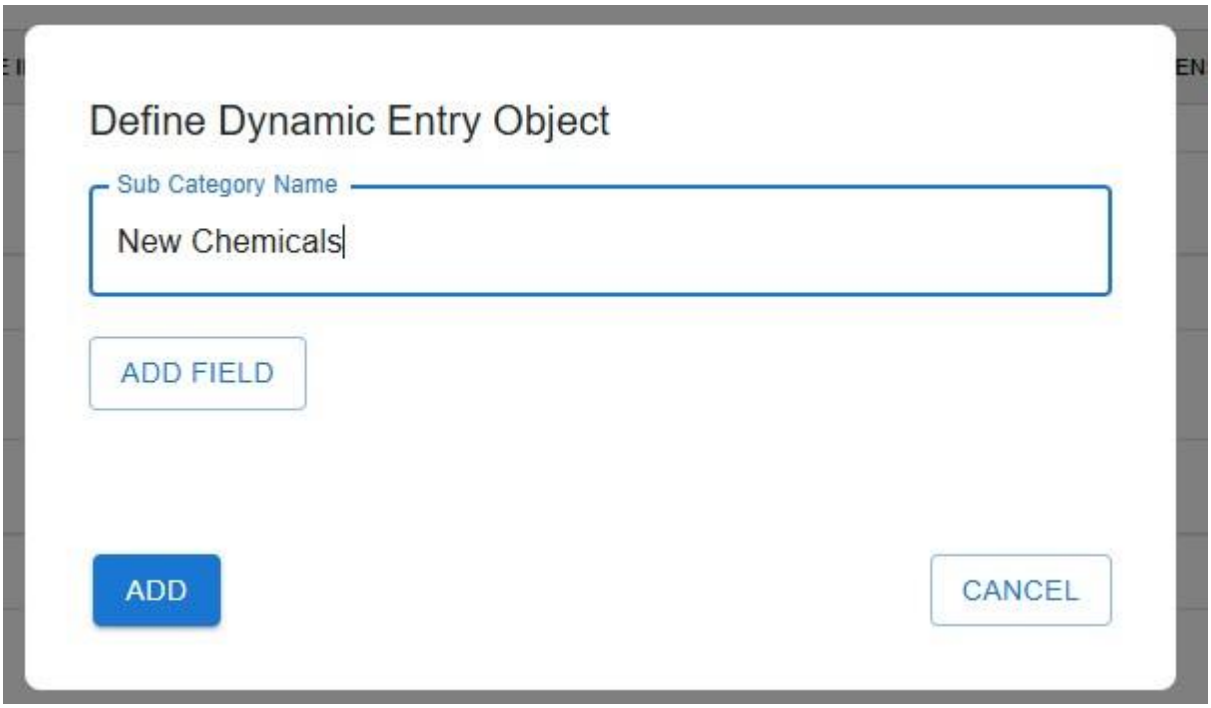
- Add new Sheets

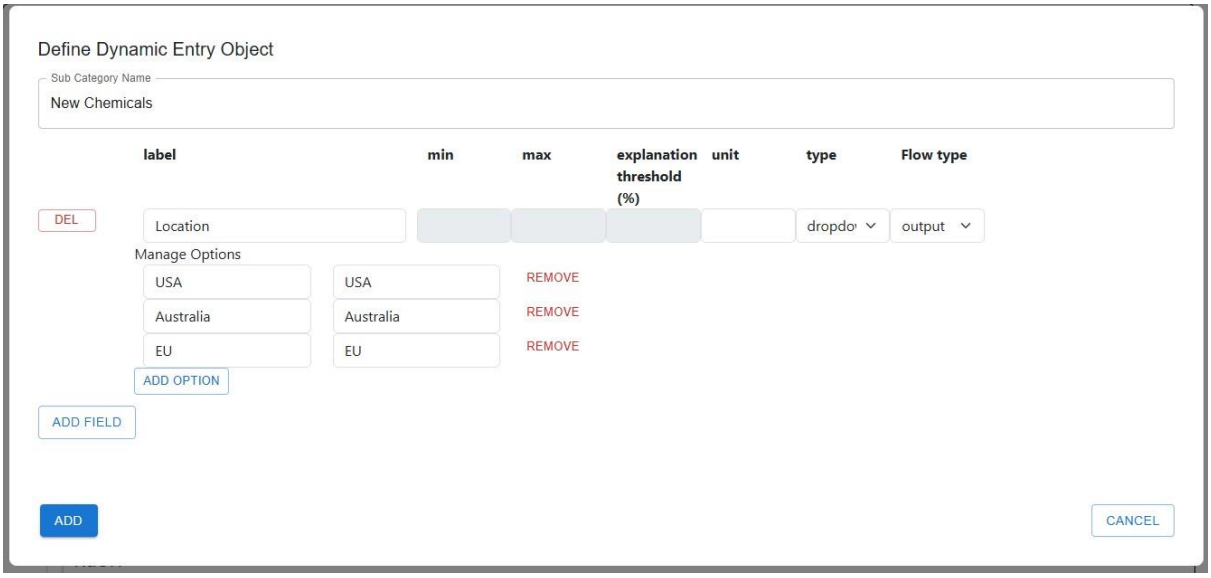
- Remove existing sheets

A.6 Allow users to add dynamic entries

The portal supports data request creators/template creators to define whether the data request receiving party can add new entries instead of just filling out the static template they received. Once the data request creators define a dynamic object, the receiving party can add more entries using the dynamic object template defined by the data request creating party.

1. Toggle Allow users to add entries switch → Edit
2. Provide appropriate subcategory name and then press ADD FIELD button to define the Dynamic entry object.

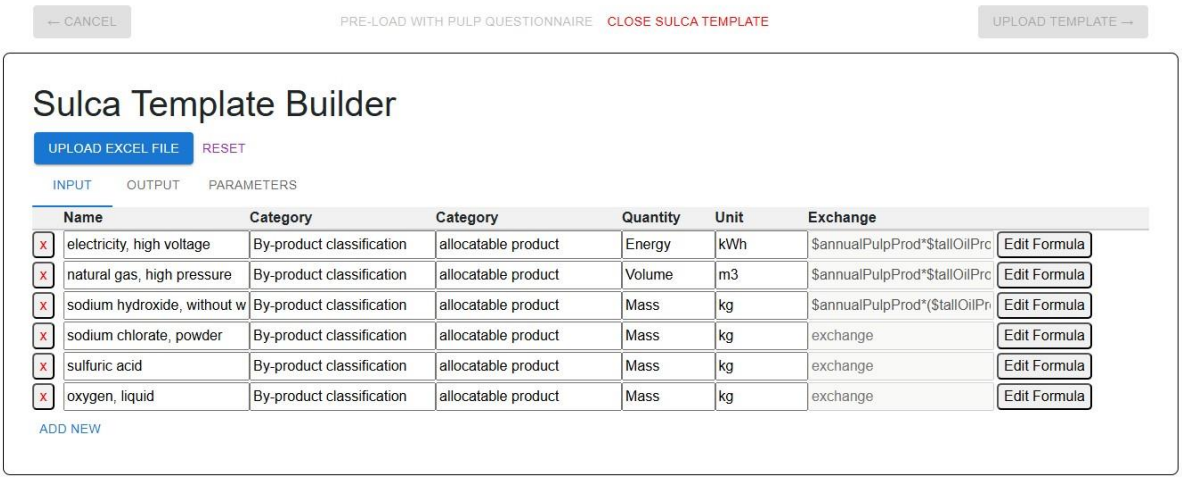




3. Once finished defining the fields → ADD
4. Then, upload the template.
5. Now, on the end user's side, they can add multiple new entries (e.g., new chemicals) by clicking the **Add New Chemicals** button.
    a. Add New Chemical → Provide an appropriate label(eg: CH4)

A.7 Sulca Conversion Template

Data request creators can define a SULCA template that specifies how they will use the data submitted by end users to generate a SULCA-compatible Excel file. By defining this template along with the original questionnaire template, data request creators can automatically download an Excel file as soon as end users submit their responses, without needing any intermediate steps to generate a SULCA-compatible file.



The following options are available for building the SULCA template.

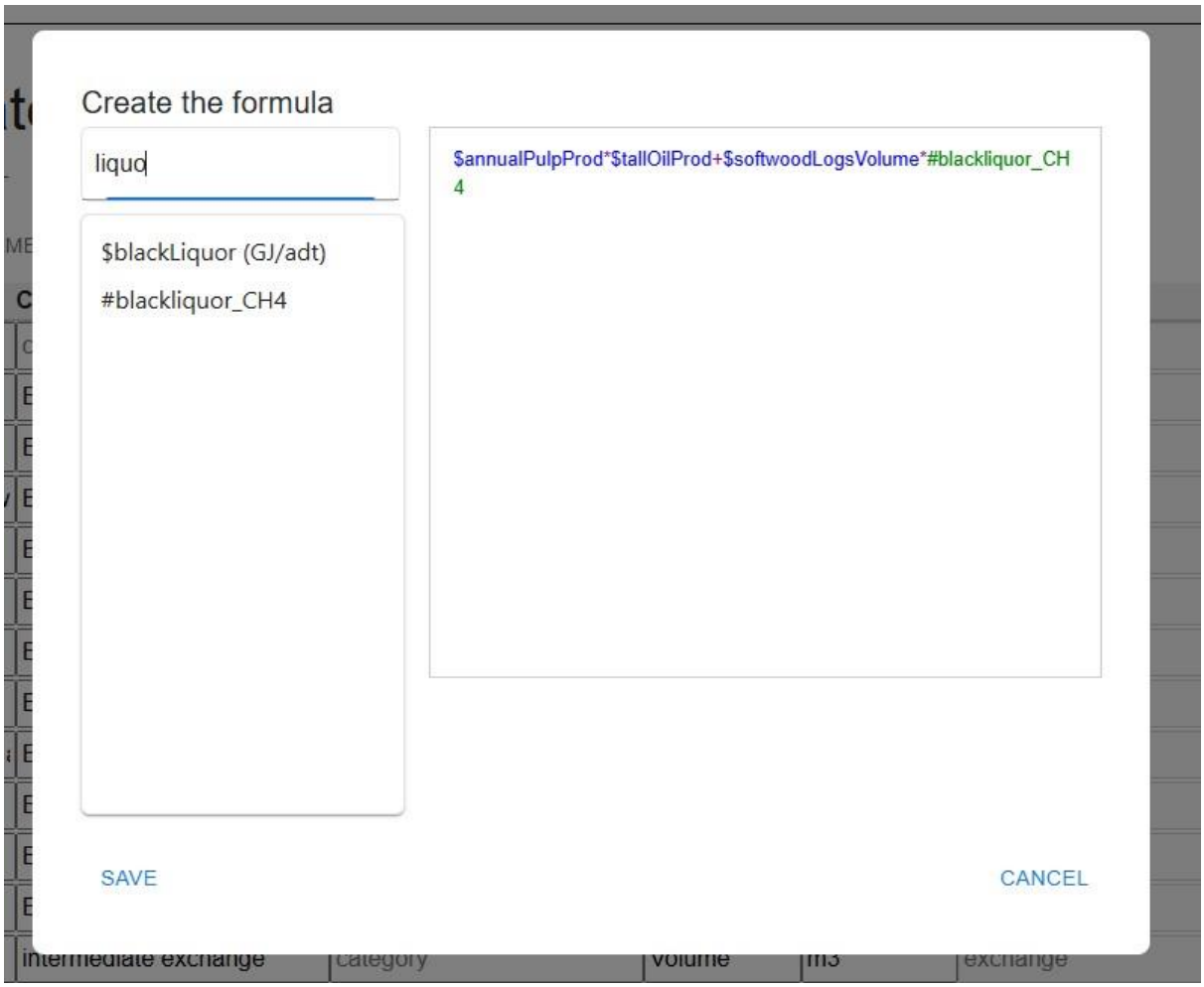
Upload Excel File	- to upload an existing excel file
Reset	- to reset the table
Add new	- to add new entries
Input tab	- Input flow entries
Output tab	- Output flow entries
Parameters tab	- to define custom variables

A.8 Add new Parameters

Users can define their own custom variables using the parameter tab.

A.9 Formula Builder

1. Sulca Template Builder → Add new → Edit Formula



Formula Builder provides the following capabilities:

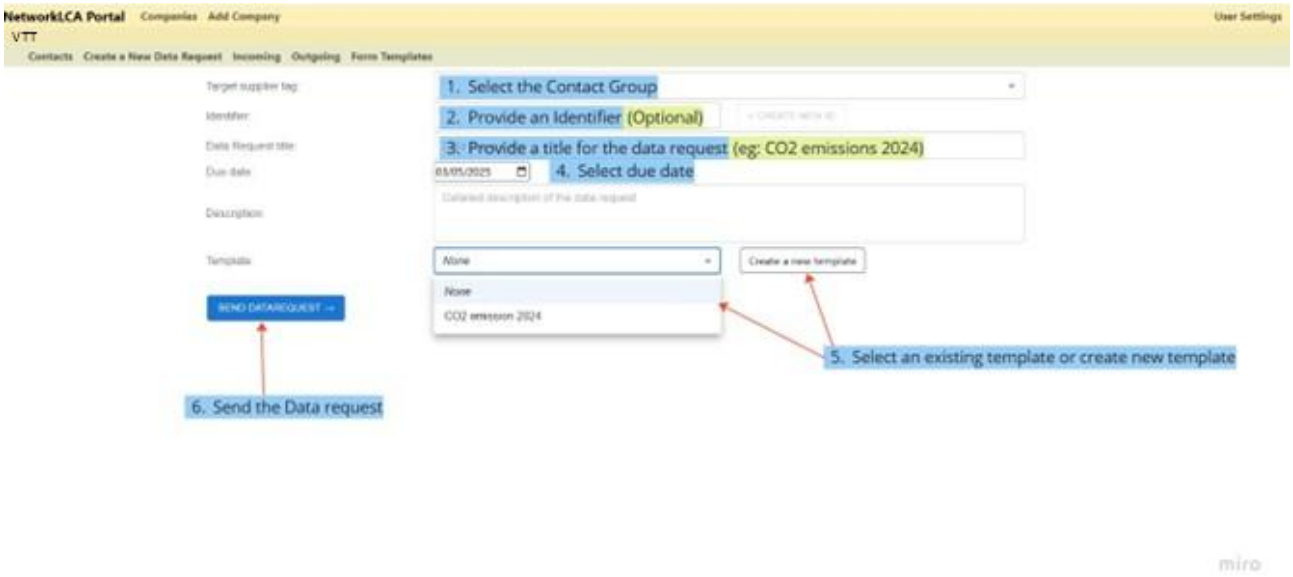
- **Search bar** – Search, find, and add variables to the equation. The search bar filters words based on any combination you type, even if you don't remember the exact variable name.
- **Formula editor** – Build and edit equations.

- You can add custom-defined parameters to the equation by searching for the parameter key.
- Parameters appear in **green**, starting with #, while **DBIDs** appear in **blue**, starting with$.
- **Validation** –
    - When you accidentally type a variable that is not in the database, it will appear in red.

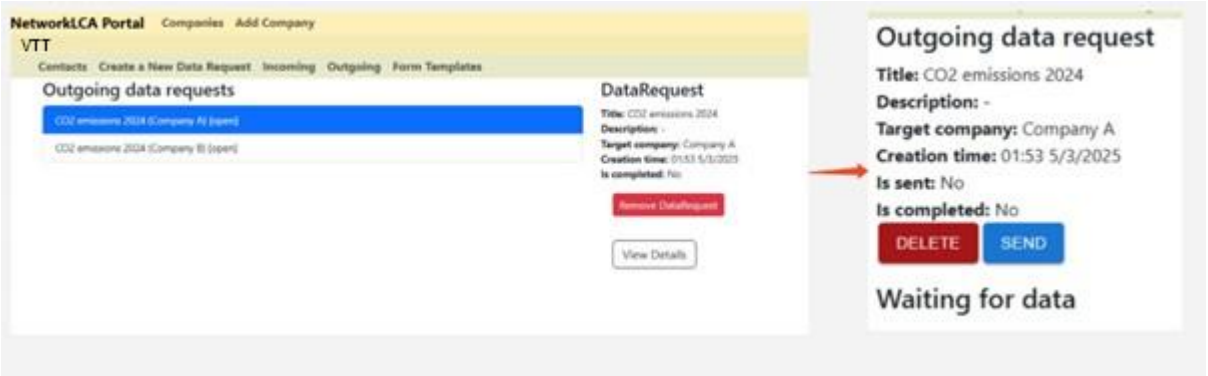
Note – When you remove a variable from the middle of an equation and try to replace it with another variable, the current version does not support directly clicking and replacing a variable from the variable list. Instead, you need to search for the variable name in the list and manually type it where it needs to be replaced. This issue only occurs when modifying an already built equation.

A.10 Create a Data Request

1. **Select Company (e.g., "VTT") → View Details**
2. Select the tab **"Create a New Data Request"** 3. Follow the steps below to create a data request.

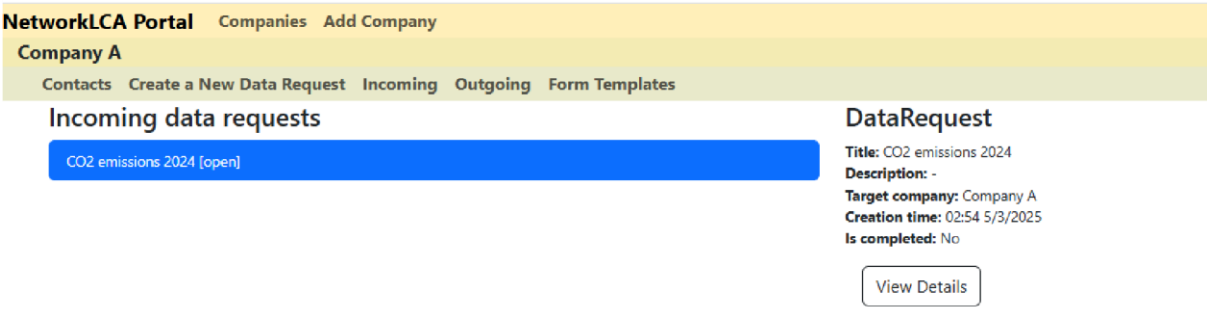


4. Go to the **Outgoing** Tab
5. Select the data request that needs to be sent(eg: CO2 emissions 2024 (Company A) [open]) → **View Details**
6. **Send** the data request or delete it if needed.



A.11 Fill in an Incoming Data Request – End Users

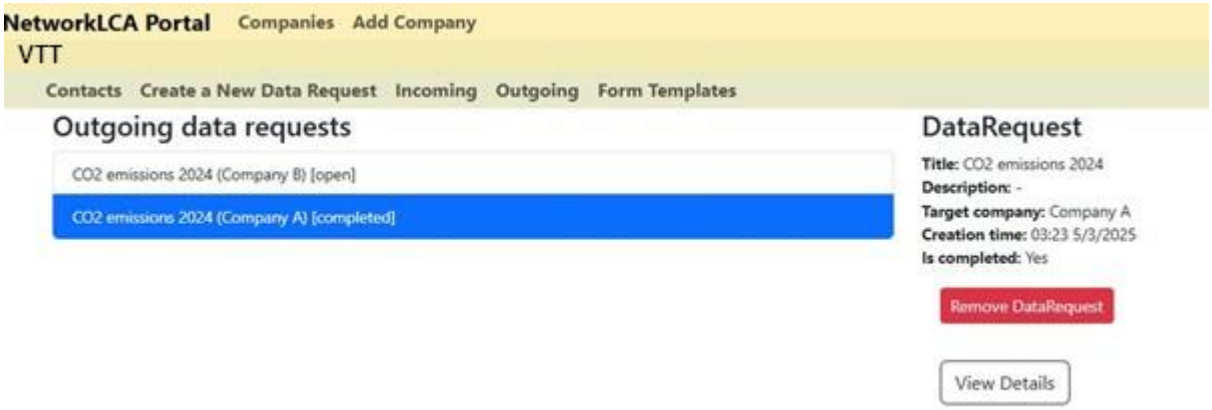
1. Select the Company(eg: Company A) → **View Details**
2. Go to **Incoming** Tab
3. Select the data request that needs to be filled → **View Details**

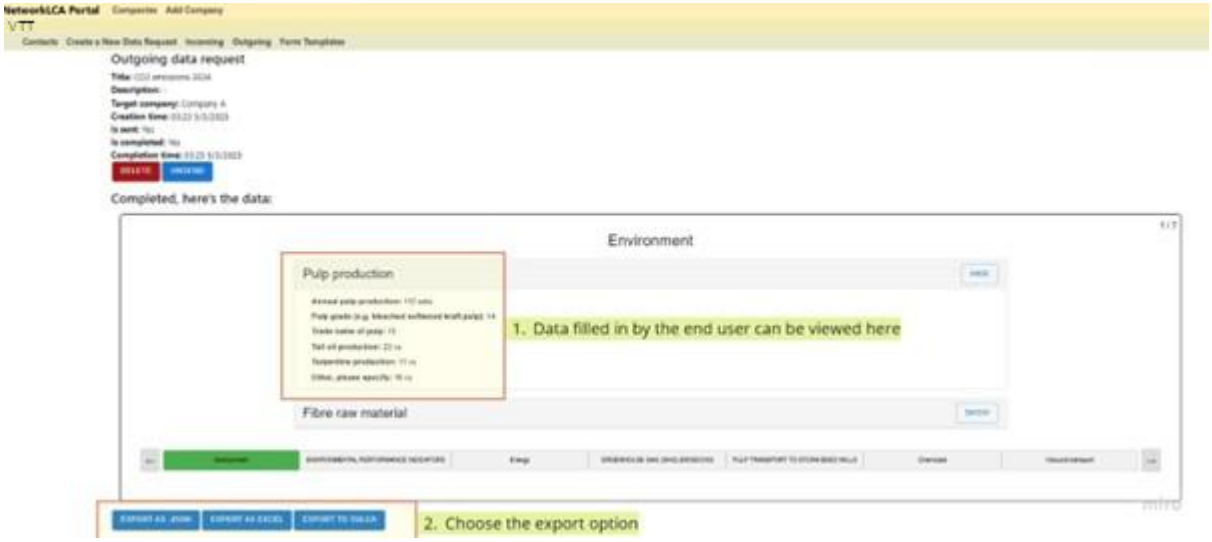


4. **Fill out the form**
5. Click **Submit** → then **Mark as Completed**

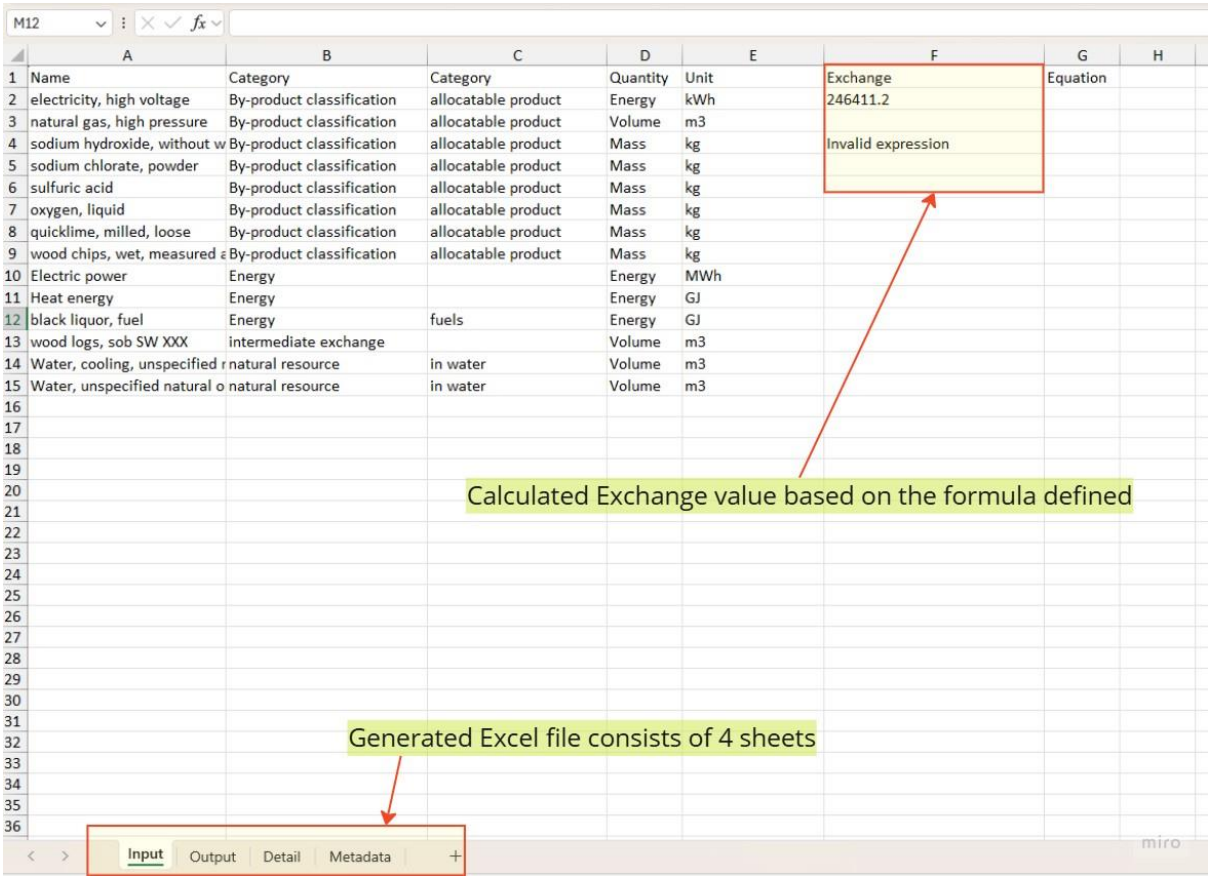
A.12 Generate a SULCA-Compatible Excel from a Completed Data Request

1. Select completed Data request from the list





2. Choose **Export to Sulca**



	A	B	C	D	E	F	G	H
1	Name	Category	Category	Quantity	Unit	Exchange	Equation	
2	electricity, high voltage	By-product classification	allocatable product	Energy	kWh	246411.2		
3	natural gas, high pressure	By-product classification	allocatable product	Volume	m3			
4	sodium hydroxide, without w	By-product classification	allocatable product	Mass	kg	Invalid expression		
5	sodium chlorate, powder	By-product classification	allocatable product	Mass	kg			
6	sulfuric acid	By-product classification	allocatable product	Mass	kg			
7	oxygen, liquid	By-product classification	allocatable product	Mass	kg			
8	quicklime, milled, loose	By-product classification	allocatable product	Mass	kg			
9	wood chips, wet, measured a	By-product classification	allocatable product	Mass	kg			
10	Electric power	Energy		Energy	MWh			
11	Heat energy	Energy		Energy	GJ			
12	black liquor, fuel	Energy	fuels	Energy	GJ			
13	wood logs, sob SW XXX	intermediate exchange		Volume	m3			
14	Water, cooling, unspecified r	natural resource	in water	Volume	m3			
15	Water, unspecified natural o	natural resource	in water	Volume	m3			

	A	B	C	D
1	Name	Formula	ExchangeWithValues	
2	electricity, high voltage	$annualPulpProd*($tallOilProd+#blackliquor_CH4)*100	112*(22+0.001)*100	
3	natural gas, high pressure			
4	sodium hydroxide, without water, in	$annualPulpProd*($tallOilProd+#blackliquor_CH4*100	112*(22+0.001*100	
5	sodium chlorate, powder			
6	sulfuric acid			
7	oxygen, liquid			
8	quicklime, milled, loose			
9	wood chips, wet, measured as dry m			
10	Electric power			
11	Heat energy			
12	black liquor, fuel			
13	wood logs, sob SW XXX			
14	Water, cooling, unspecified natural			
15	Water, unspecified natural origin			
16	Carbon dioxide, fossil			
17	Carbon dioxide, non-fossil			
18	Carbon monoxide, fossil			

A.13 Export options

Export as JSON – Generate a JSON file.

Export as EXCEL – Generate an Excel file with all the data fields and attributes.

Export to SULCA – Generate a SULCA-compatible Excel file with a custom-added metadata sheet and an additional exchange column to print the calculated exchange value based on the defined formula.

A.14 Sulca Compatible Excel

Consists with 4 sheets

Input	Defined input and output flows, including an additional exchange column with calculated exchange values based on the defined formulas.
Output	
Detail	The General Details sheet consists of the template name.
Metadata	A custom-added sheet, other than the defined SULCA upload Excel format, prints metadata such as the formula (defined by the data request creator) and the exchange values (the same formula, but with the values from the completed data request)

Formula Validation and error messages

1. **Expression errors** – such as missing parentheses.
2. **databaseId $(dbid) not found** – when an incorrect dbID is typed in the formula.