EuroSun 2026

AN INTERACTIVE TOOL FOR OPTIMIZING SUSTAINABLE BUILDING ENERGY SYSTEMS

Summary

A free graphical user interface for energy system modelling was developed. The browser based tool consists of a front end and a back end. In the front end, the energy system model is created on a digital whiteboard and then transferred to the back end. In the back end, the energy system is transferred to oemof.solph [5] and solved using Pyomo, a Python-based open-source optimization modeling language. The results are then processed and visualised. The open tool has proven to be extremely suitable for people who do not have extensive programming experience. The browser-based solution offers the advantage that no software needs to be installed for the development of energy systems in the building sector.

Keywords: Decision making–Mathematical models, energy modelling, building sector, graphical user interface, oemof, open source

Field of application for the findings: Optimization and decarbonization of energy systems in the building sector

Addressed audience: Engineers and building designers

1. Introduction

The transition to a climate-neutral energy system poses a significant challenge. One way to support this transition is through the use of energy system models. The open-source modelling framework (oemof) [3] is one example of a generic modelling environment. Oemof is a tool / framework developed using Python, which requires certain programming skills to use. As a result, access to such tools / modelling frameworks is difficult for a large number of users. This problem has been recognised by several institutes, which have developed appropriate solutions. The tool presented below enables the mapping of energy systems and optimisation in terms of costs and CO_2 emissions. Since a significant proportion of energy demand is attributable to the heating sector, there is considerable potential for optimising systems in this sector. In view of this, the tool presented can be considered a potentially valuable instrument for supporting the process of converting energy supply in buildings.

2. Methodology

The tool developed is a browser-based application divided into a front end and a back end. A key advantage of the browser-based approach is that no additional software needs to be installed on user's computer. It should also be noted that the computing capacity of the user devices is not a significant factor.

2.1 Front End

The front end of the platform acts as the primary interface for modeling and simulating energy systems. The tool enables users to create, configure, and evaluate energy scenarios in an interactive, graphical environment. In this context, the computational complexity is abstracted for users.

The angular framework [4] based front end is designed to ensure optimum user-friendliness while maintaining technical completeness. It simplifies complex modelling processes through structured configuration forms, predefined system components and guided scenario management, while still supporting advanced parametric and investment-based modelling approaches. The separation of user interaction from backend calculation ensures responsiveness, scalability and efficient handling of simulation processes through the front end.

2.2 Back End

The backend is based on a modern microservice-oriented architecture, using the FastAPI framework [6] to provide REST functionality.

The implementation follows the model-view-controller pattern, with a clear separation between routing, data models and logic. The technological basis of the system is Python with type hints, which serve to optimise code quality. FastAPI is used as a powerful, asynchronous web framework with automatic OpenAPI documentation. SQLModel [7] is used as the ORM (object-relational mapping) layer for data persistence. This enables type-safe database interactions based on Pydantic [2] and SQLAlchemy [1]. The relational database allows flexible storage of complex energy system configurations.

The entire system is containerised using Docker Compose. So it is easy to start the project on own hardware, known as on-premise. The code itself is stored in a public GitHub-repository under the aGPLv3 License.

3. Building-Template

The generic approach of the tool presented proves to be extremely suitable for developing a general approach to modelling energy systems, such as buildings. These can comprise sources, sinks, converters, storage devices and buses, which can be connected via drag-and-drop. The general approach is characterised by the fact that it has no technology-limiting restrictions. However, a thorough examination of the oemof.solph-documentation is necessary for a precise description of the components. To simplify working with the tool, various templates have been provided that can be used to preconfigure energy systems. This makes it easier to get started with simulation using the tool. In this context, a template for building simulation has been developed, which will be presented in detail below.

The front end is set up on a project basis. You can either create a new, empty project or use an existing template as a basis. At the project level, you can define global settings, including the project name, the units used and the location. Within a project, you can generate and manage different scenarios. For example, an existing scenario and various technology scenarios were created for the building project template.

In a technology scenario, for example, the use of a heat pump is analysed. In addition to the heat pump, conventional technologies such as gas boilers are also provided. The optimiser selects the most cost-effective technology based on stored investment and operating costs and returns the results to the user. The energy system created in the tool is visualised in Figure 1 below.

Figure 1 shows various technology components: an electrical grid connection, a conventional electrical load that can be stored with load profiles, and a photovoltaic system that is also set up as a source block. It is already apparent here that the generic approach using sources is used for two different applications. In addition, a battery storage system has been implemented to compensate for differences between supply and demand. All these components are connected to the electricity bus. An air-to-water heat pump is used as the sector coupling technology. A converter is used as a component for this purpose. On the output side of the heat pump is the heat bus, which acts as a busbar for all energy flows in the thermal sector. A gas boiler could also provide heat here. In the thermal sector, the system is also assigned a storage facility and a specific load profile.

Additional components can be easily assigned to the energy system via drag-and-drop and edited by double-clicking or right-clicking. The individual components can include time series such as variable electricity prices, feed-in profiles and load profiles. These time series can be used in different time resolutions. Typically, however, hourly or quarter-hourly time series are used.

4. Results

Once a scenario has been created, it can be simulated. The results enable the visualisation of energy flows in temporal resolution and the representation of the cost-optimised performance of the individual components. This approach can support decision-making.

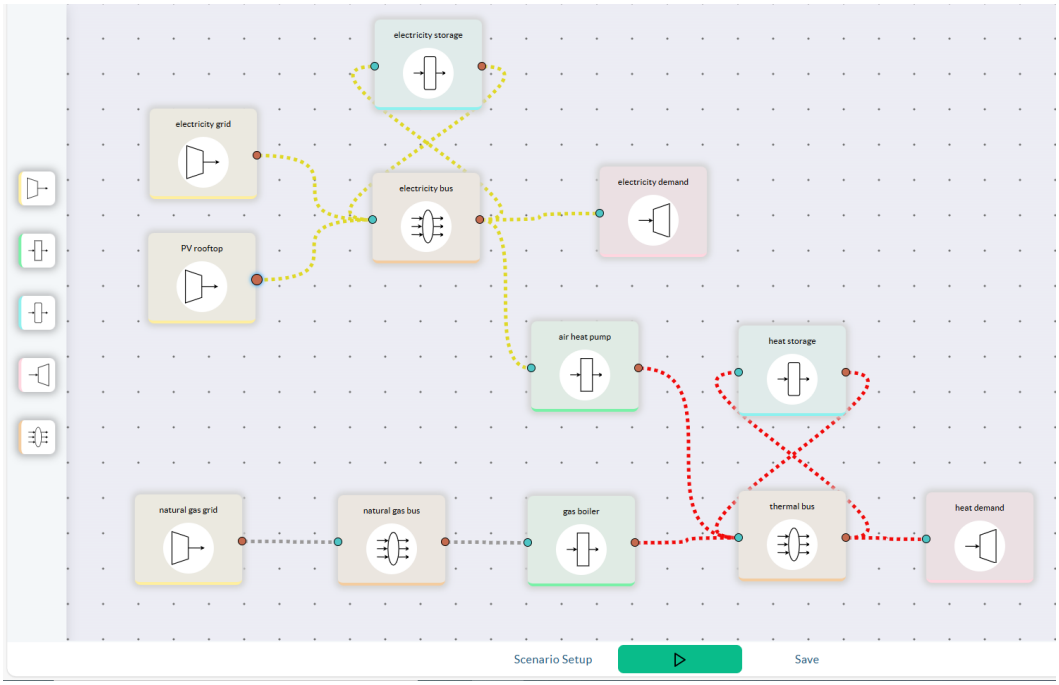

Fig. 1: Workspace of the graphical user interface.

5. Presentation

Parts of the presentation are planned as live demonstrations.

References

[1] M. Bayer. The Architecture of Open Source Applications (Volume 2). lst accessed on 25/02/2026. URL: https://aosabook.org/en/v2/sqlalchemy.html.

[2] S. Colvin, E. Jolibois, H. Ramezani, A. Garcia Badaracco, T. Dorsey, D. Montague, S. Matveenko, M. Trylesinski, S. Runkle, D. Hewitt, A. Hall, and V. Plot. Pydantic Validation. lst accessed on 25/02/2026. URL: https://docs.pydantic.dev/latest/.

[3] S. Hilpert, C. Kaldemeyer, U. Krien, S. Günther, C. Wingenbach, and G. Plessmann. The Open Energy Modelling Framework (oemof) - A new approach to facilitate open science in energy system modelling. *Energy Strategy Reviews*, 22:16–25, 2018. https://doi.org/10.1016/j.esr.2018.07.001 doi:10.1016/j.esr.2018.07.001.

[4] Nilesh Jain, Ashok Bhansali, and Deepak Mehta. AngularJS: A modern MVC framework in JavaScript. In *Journal of Global Research Volume 5*, volume 12, pages 17–23.

[5] Uwe Krien, Patrik Schönfeldt, Jann Launer, Simon Hilpert, Cord Kaldemeyer, and Guido Pleßmann. oemof.solph—A model generator for linear and mixed-integer linear optimisation of energy systems. *Software Impacts*, 6:100028, 2020. https://doi.org/10.1016/j.simpa.2020.100028 doi:10.1016/j.simpa.2020.100028.

[6] S. Ramírez. FastAPI Framework. lst accessed on 25/02/2026. URL: https://fastapi.tiangolo.com.

[7] S. Ramírez. SQLModel. lst accessed on 25/02/2026. URL: https://sqlmodel.tiangolo.com/.