HarvRESt
Greener Farming with RES

D5.2

RES technologies library

01 / 09 / 2025

Funded by
the European Union

www.harvrest.eu

PROJECT INFORMATION

ACRONYM	HarvRESt
PROJECT NAME	Harnessing the vast potential of RES for sustainable farming
PROGRAMME	Horizon Europe
TOPIC	HORIZON-CL6-2023-CLIMATE-01-7
TYPE OF ACTION	HORIZON Research and Innovation Actions
PROJECT NUMBER	101136904
START DAY	1 January 2024
DURATION	36 months

DOCUMENT INFORMATION

TITLE	D5.2 RES technologies library
WORK PACKAGE	WP5
TASK	T5.2
LEAD PARTNER	CIRCE
CONTRIBUTORS	-
DATE	01 / 09/ 2025
DISSEMINATION LEVEL	Public

DOCUMENT HISTORY

VERSION	DATE	CHANGES	RESPONSIBLE PARTNER
0.1	19/11/2024	TOC definition	CIRCE
0.2	17/12/2024	First review models (SOA)	CIRCE
0.3	25/02/2025	RES models Included	CIRCE
0.4	22/04/2025	Demand and Meteorological data models included	CIRCE
0.5	19/06/2025	Agricultural models included	CIRCE
0.6	08/08/2025	1st draft	CIRCE
0.7	22/08/2025	Review by SUITE5	SUITE5
1.0	01/09/2025	Reviewed and consolidated version	CIRCE

TABLE OF CONTENTS

ABBREVIATIONS

AC	Alternating Current
ADM1	Anaerobic Digestion Model No. 1
API	Application Programming Interface
APSIM	Agricultural Production Systems Simulator
AVPP	Agricultural Virtual Power Plant
BDEW	Bundesverband der Energie- und Wasserwirtschaft (Federal Association of the Energy and Water Industries in English)
BiPMo	Biogas Plant Model
BSM2	Benchmark Simulation Model No. 2
CAPEX	Capital Expenditures
CCGT	Combined Cycle Gas Turbine
CHP	Combined Heat and Power
CLI	Command Line Interface
CSV	Comma Separated Values
DC	Direct Current
DHI	Diffuse Horizontal Irradiance
DNI	Direct Normal Irradiance
DSSAT	Decision Support System for Agrotechnology Transfer
DTU	Technical University of Denmark
ERA5	Fifth generation from European Centre for Medium-Range Weather Forecasts reanalysis for the global climate and weather for the past 8 decades
ESMAP	Energy Sector Management Assistance Program
ET_0	Reference Evapotranspiration
EUPL	European Union Public License
FAO UN	Food and Agriculture Organisation of the United Nations
GHI	Global Horizontal Irradiance
GIS	Geographic Information System
GUI	Graphical User Interface
GWA	Global Wind Atlas

H-VPPs	Hybrid Virtual Power Plants
H2Integrate	Holistic Hybrids Optimisation and Design Tool
I/O	Input / Output
JSON	JavaScript Object Notation
LAI	Leef Area Index
LCOE	Levelized Cost Of Electricity
LINGRA	Light Interception Grassland
LINTUL	Light Interception and Utilisation
LP	Linear Programming
LPG	LoadProfileGenerator
MCDA	Multi-Criteria Decision Analysis
MILP	Mixed-Integer Linear Programming
MIT	Massachusetts Institute of Technology
NREL	National Renewable Energy Laboratory
OHySEM	Optimised Hybrid Systems for Energy and Market management
OSI	Open Source Initiative
O&M	Operation and Maintenance
PASE	Python Agrivoltaic Simulation Environment
PCSE	Python Crop Simulation Environment
PV	Photovoltaic
PVGIS	Photovoltaic Geographical Information System
PyADM1	Phyton Anaerobic Digestion Model No. 1
PyPSA	Python for Power System Analysis
PySAM	Phyton System Advisor Model
PyWAsP	Phyton Wind Atlas Analysis and Application Program
RES	Renewable Energy Source
SaaS	Software as a Service
SAPM	Sandia PV Array Performance Model
SAM	System Advisor Model
SDK	Software Development Kit

SSC	SAM's Simulation Core
TMY	Typical Meteorological Year
UniSQ	University of Southern Queensland
US	United States
USGS	United States Geological Survey
WOFOST	World Food Studies
WRF	Weather Research and Forecasting
YAML	Yet Another Markup Language

1. EXECUTIVE SUMMARY

This deliverable presents the outcomes of Task 5.2 of the HarvRESt project, establishing the technical foundations and tools required for developing the Agricultural Virtual Power Plant (AVPP) in Task 6.1, as a complement to Task 5.3 and a continuation of Task 5.1. The document compiles, analyses, and organises a library of renewable energy source (RES) technologies and associated modelling tools suitable for integration into agricultural environments, with a strong focus on open-source resources and Python-based solutions to ensure transparency, interoperability, and long-term adaptability.

The work begins with a state-of-the-art review of available tools for RES generation and storage modelling, agricultural production modelling, demand modelling, and hybrid system simulation. Each tool is described in terms of its licence, required inputs, internal modelling structure, main outputs, and potential integration within the AVPP framework. The analysis prioritises solutions capable of representing multiple RES technologies (e.g., solar photovoltaic, wind, hydro, biomass, storage) and their coupling with agro-specific models, enabling realistic simulations of energy–agriculture interactions.

The assessment includes prominent open-source libraries such as pvlib, windpowerlib, HydroGenerate, PySAM, and selected agronomic modelling platforms, such as APSIM, alongside other specialised tools for hybrid systems, energy management, and demand forecasting. Proprietary solutions (e.g., HOMER) are also considered where relevant, with integration pathways proposed despite licensing constraints. For each category, strengths, limitations, and identified gaps are reported, highlighting where custom development may be required to meet HarvRESt's specific needs.

Building on this review, the deliverable defines at a conceptual level the RES Technologies Library — a modular, functional repository to host models, functions, and data structures forming the computational core of the AVPP. This conceptual framework outlines how technology-specific modules (e.g., PV generation, wind resource modelling, crop growth impacts) will be standardised and interconnected, supporting hybrid energy–agriculture simulations, optimisation routines, and scenario analyses.

The proposed library will serve as both a reference for developers and a practical integration layer, enabling AVPP to combine RES performance modelling with agricultural productivity assessments, demand profiles, and decision-support capabilities. By mapping current capabilities and future development needs, this work creates a bridge between earlier analytical phases and the implementation of the AVPP, providing a robust basis for delivering climate-neutral, energy-efficient, and scalable solutions for European agro-communities.

2. INTRODUCTION

This deliverable presents the outcomes of Task 5.2, which focuses on the identification, compilation, and technical characterisation of existing modelling tools for renewable energy technologies applicable in agricultural settings. The task aims to gather, evaluate, and organise open-source libraries and modelling resources that are relevant to the simulation of renewable energy technologies in agricultural environments.

The goal is to build a technically documented and accessible library of tools that can support the design, simulation, and optimisation of hybrid energy systems in agricultural contexts. These tools will also provide the foundation of the modelling and integration activities in the AVPP.

In addition, agronomic modelling tools are reviewed, to address the specific needs of farming operations. The goal is to support decision-making and optimisation of RES integration within agricultural settings, ultimately contributing to the development of climate-neutral farms.

By linking the tools to the technologies, data and needs of the use cases, this deliverable facilitates a more accurate and context-aware modelling process, while also promoting transparency, interoperability, and replicability across different farming scenarios.

2.1 Goals

The main goals of this document are:

- Identify and classify a set of existing software libraries and modelling tools relevant for RES integration and hybrid energy system simulation;
- Describe for each tool its required inputs, internal modelling structure and typical outputs;
- Provide a state-of-the-art review of modelling tools relevant to AVPP design, drawing on the technologies identified in WP2 and use case-specific data;
- Compile and present the selected modelling resources within a structured reference library, serving as an illustrative collection of tools identified and/or developed during the task. This library will also outline a proposed structure for an internal repository to support the future development of functions based on existing or ad-hoc models, aligning with the architecture underpinning the AVPP and facilitating their progressive integration; and
- Highlight gaps or limitations, in existing tools, which may need to be addressed in future project phases or through custom developments.

This deliverable is therefore a bridge between the early analytical and data-gathering phases of the project and the modelling and optimisation efforts that follow. It consolidates valuable information for decision-making and technical implementation, laying a solid foundation for building climate neutral, energy-efficient, and scalable solutions in European agro-communities.

3. REVIEW OF TOOLS AND TECHNOLOGIES

The following section presents a review of software libraries that could fit within the framework of the project's development. For each type of renewable source or component of interest to be integrated into the subsequent development of the AVPP, an analysis is provided of their functionalities, input requirements, outputs, and feasibility of integration into the project.

The analysis aims to identify suitable tools, preferably those with open-source licensing, so that they can align with the public dissemination level of the deliverable and be integrated without barriers into future project developments. The focus will be on finding tools built in Python, which will serve as the base development language for the AVPP.

This review focuses on tools that could potentially be used within the AVPP framework. Some of these tools may later be integrated into the AVPP through an internal library that will be developed to support its execution. This function library will evolve throughout the HarvRESt (AVPP) project and may serve as the foundation for similar future developments, including extended functionalities. The conceptual definition and structure of this internal library will be presented in 0.

3.1 RES modelling

3.1.1 *Solar PV Energy*

Pvlib

Pvlib python [1], [2] is an open-source library developed by a global community to provide a comprehensive set of functions and classes for simulating the performance of photovoltaic (PV) energy systems. Its primary mission is to offer open, reliable, interoperable, and benchmarked implementations of PV system models, making it an invaluable tool for both researchers and industry professionals in the solar energy sector. The source code is published under the permissive BSD 3-Clause license, allowing unrestricted commercial and academic use provided attribution and warranty disclaimers are retained.

The library originated as a Python translation of the PV_LIB toolbox for MATLAB, which was initially developed by Sandia National Laboratories. Since then, pvlib python has expanded significantly, incorporating contributions worldwide. It is actively maintained by a core group of PV modelers from various institutions, ensuring that it stays up-to-date with the latest developments in PV technology and modelling techniques.

Pvlib python (Figure 1) covers a wide range of functionalities essential for PV system simulation and analysis. These include modelling solar position, irradiance, atmosphere conditions, and the electrical performance of PV modules and systems. By leveraging powerful Python libraries like NumPy and Pandas, pvlib python enables efficient data manipulation and numerical computations, facilitating complex simulations and analyses.

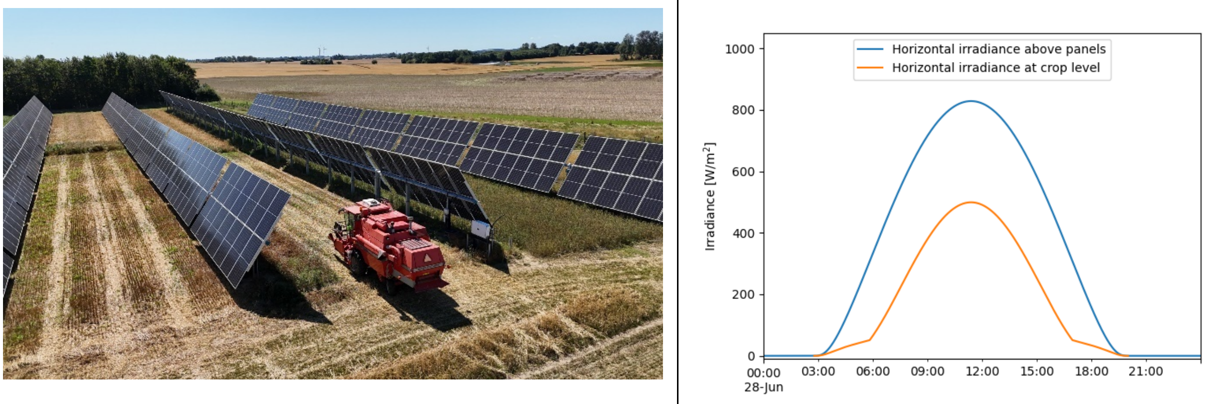

Figure 1. Pvlib examples [55]

The library expects two distinct classes of input. Intrinsic parameters that describe the system—module electrical coefficients, inverter efficiency curves, array geometry, mounting type and location metadata—are bundled in a PVSystem (and its Array) object. Extrinsic data that drive the system—time-stamped vectors of global, direct and diffuse irradiance, ambient temperature, wind speed or pre-calculated cell temperature—are supplied at runtime as a pandas.DataFrame to one of the ModelChain.run_model methods. The same weather frame can be broadcasted to all arrays or provided separately for each array, supporting multi-array systems with heterogeneous orientations.

Internally pvlib organises the calculation steps into a ModelChain, an interface that wires together the selected sub-models for solar position, irradiance transposition, airmass, spectral modifiers, angle-of-incidence losses, cell temperature, DC conversion (single-diode, PVWatts, SAPM, etc.), inverter conversion and optional loss models. Each link in the chain is a pure-Python function that can be swapped or extended, and the chain automatically infers sensible defaults from the data attached to the PVSystem when the user does not make an explicit choice. This modular architecture keeps each algorithm testable in isolation while still exposing a one-line "run" interface for production workflows.

The chain returns as output a "ModelChainResult" container whose attributes are time-series pandas objects: plane-of-array irradiance components, angle-of-incidence and spectral loss factors, cell temperatures, detailed single-diode currents and voltages, DC power and final AC power.

Application and integration within the project

The pvlib library could be highly useful within the project's development environment to refine the irradiance inputs to match the specific characteristics of the PV installation, calculate the solar position, and parametrise the PV system. This would allow for accurate solar production estimates based on the available solar resource (solar radiation), ensuring a tailored and efficient modelling process for the PV plant. Furthermore, it could be useful in generating AVPP scenarios by adjusting the parameters of the parametric model, enabling the evaluation of different outcomes when modifying the plant size or the type of installation (e.g., fixed-axis, single-axis tracker).

3.1.2 *Wind energy*

Windpowerlib

Windpowerlib is a lightweight, Python-based library [3] for estimating the electrical output of individual wind turbines or entire farms from standard meteorological inputs and turbine specifications (Figure 2). It implements a modular, object-oriented “ModelChain” workflow—covering height and air-density corrections, power-curve interpolation, and optional wake-loss adjustments—to produce time-series feed-in estimates. It is distributed under MIT License and installable via pip, it can be embedded as a generic wind-power module in broader renewable-energy simulation environments.

The required inputs comprise: the time series of wind speeds (at one or more heights), air temperature, surface roughness length and air pressure; turbine specifications, i.e. power curve or power-coefficient curve, nominal power, hub height, rotor diameter. Data can come from the built-in OpenEnergy Database or be provided by the user. Optionally, farm/cluster definitions can be provided, such as layout, inter-turbine distances and wake-loss parameters for modelling farms or clusters.

The windpowerlib internal modelling structure undergoes: a height and density correction, by adjusting raw wind speeds and air density to turbine hub height; power output calculation, by interpolating turbine power or coefficient curves to compute instantaneous output; wake-loss and smoothing, by applying wake-loss models or curve-based derating plus optional smoothing functions; and a final aggregation for farms/clusters, gathering individual turbine outputs with wake interactions via the WindFarm or WindTurbineCluster classes. The outputs are corrected wind speed, air density, raw and wake-adjusted power outputs, with access to corrected inputs and intermediate results for diagnostics.

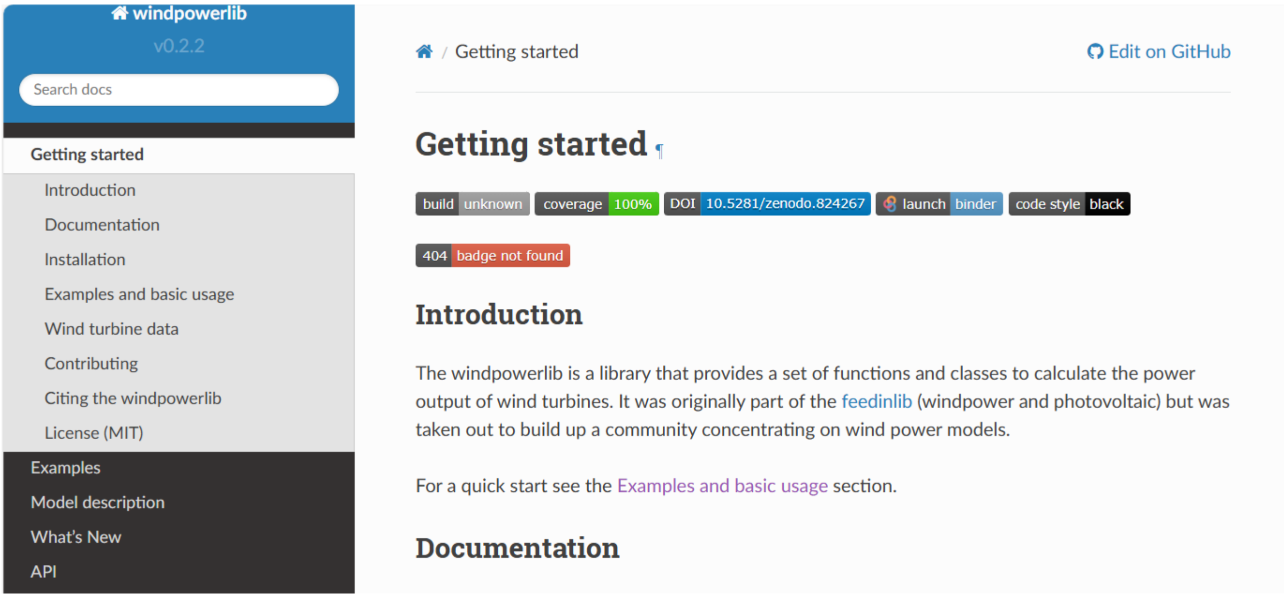

Figure 2. Windpowerlib [56]

Application and integration within the project

As a pure Python, pip-installable library with minimal dependencies and an MIT license, windpowerlib can be seamlessly embedded in larger energy-system frameworks or custom simulation tools as the concept of the AVPP sets. Its simple API, by instantiating a ModelChain with weather and turbine definitions, and retrieving a feed-in DataFrames, makes it ideal as a modular wind-power estimator within comprehensive renewable-energy modelling environments. It is worth noting that domestic-scale turbines, which might be installed in agricultural settings, can pose a challenge. However, provided accurate small-turbine power curves and local wind data are available, windpowerlib's height-correction and power-curve interpolation modules apply unchanged, enabling basic performance estimates. Simple wake-loss derating can also be used for small clusters of units. However, because urban turbulence and building-induced flow effects lie outside the library's semi-empirical wake models, rooftop or complex-terrain installations should be paired with higher-resolution wind inputs or specialised flow models to capture site-specific flow dynamics accurately.

3.1.3 *Hydro energy*

HydroGenerate

HydroGenerate [4] is a Python-based library developed by Idaho National Laboratory for estimation of hydropower generation time series from site-specific flow and hydraulic head data. It supports both user-provided flow inputs and direct retrieval from USGS water data services, automatically selects or computes turbine efficiency curves based on head or turbine type, and outputs detailed performance and cost metrics via a simple API (Figure 3). HydroGenerate is released under the BSD 3-Clause license, permitting free use, modification, distribution, and sublicensing provided the original copyright and license notice are retained.

The tool requires as basic inputs [5]: volumetric flow rate time series and hydraulic head (difference between intake and discharge elevations). Optional but frequently used arguments include rated power, turbine type or an automatic turbine selector, design-flow percentile, unit system, and flags governing head-loss models, maintenance scheduling, cost and revenue estimation, or hydrokinetic parameters. The detailed input schema [5] lists more than thirty parameters—among them penstock geometry, generator efficiency, electricity price, capacity factor targets and minimum environmental flows—so the model can run from quick screening all the way to techno-economic assessment without altering the core code.

The library's internal modelling structure exposes a main interface function, which handles both simple and time-series computations. Internally, it follows a general workflow: flow net head (accounting for penstock losses via Darcy-Weisbach or Hazen-Williams), turbine efficiency (varying with flow and turbine type), generator efficiency, power output and optionally techno-economic evaluation (CAPEX, O&M, LCOE, revenue) Three principal scripts—turbine_calculation.py, hydraulic_processing.py and hydropower_potential.py—are wrapped behind a single high-level function, calculate_hp_potential, that orchestrates the workflow. Each physical or economic process is represented by a lightweight class that inherits from an abstract base class, ensuring any new turbine algorithm, head-loss routine or cost model can be plugged in.

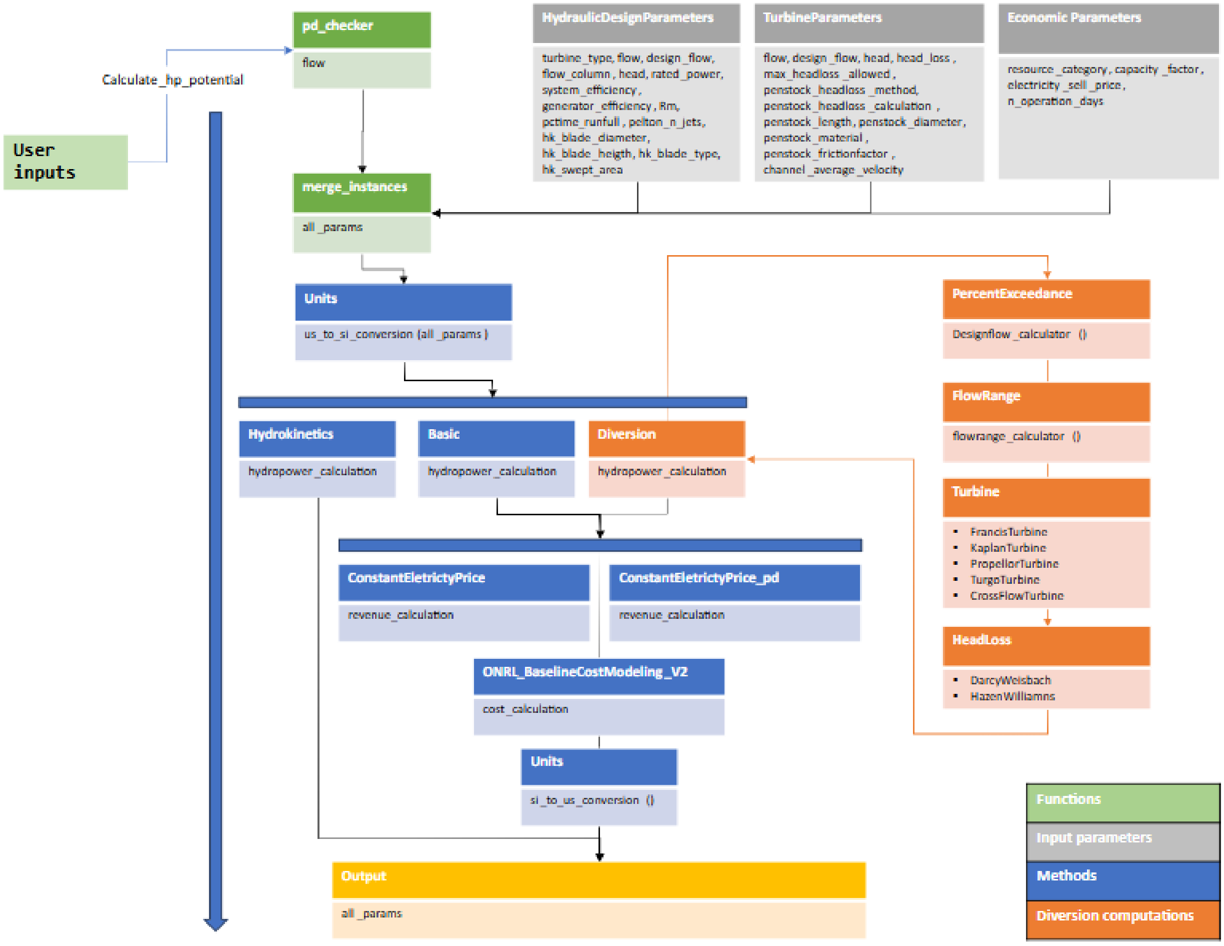

Figure 3. HydroGenerate Computations Workflow, methods, functions, and classes [57]

The tool outputs both granular and aggregated results: a power-versus-time series, turbine and overall efficiencies, flow-duration and capacity-factor curves, net and gross heads, and derived metrics such as annual energy, O&M costs, capital expenditure and revenue. Intermediate artefacts—flow-duration tables, efficiency curves, design-flow estimates, head-loss details—are also held as Pandas data-frames or NumPy arrays, ready for immediate visualisation or export.

Application and integration within the project

HydroGenerate codebase is developed in pure Python and relies only on ubiquitous scientific libraries such as NumPy, Pandas and SciPy, embedding it inside a holistic web-based energy-system simulator written in Python is technically straightforward. The BSD license ensures that a web service can bundle HydroGenerate without copyleft obligations, while the modular API lets developers call turbine sizing, head-loss or economic sub-routines independently or chain them with PV, battery or grid-dispatch models. Outputs are native Python objects that can be serialised to JSON for REST endpoints, and optional calls to the USGS API for live flow data can be stubbed or proxied to fit an offline compute environment. In short, minimal dependency overhead, permissive licensing and a clean, object-oriented design make HydroGenerate highly amenable to seamless integration within a broader renewable-energy simulation platform.

3.1.4 *Hybrid and System-Level Energy Simulation Tools*

This subsection encompasses software platforms designed to model, simulate, and optimise the performance of integrated renewable energy systems, including hybrid configurations (e.g., PV + wind + storage) and microgrids. These tools, such as HOMER and the System Advisor Model (SAM/PySAM), enable detailed techno-economic assessments, performance simulations, and feasibility analyses, making them highly valuable for evaluating renewable energy projects in realistic operational and financial contexts.

SAM (System Advisor Model) and PySAM library

The System Advisor Model (SAM) **[6]** is a free, comprehensive desktop application developed by the National Renewable Energy Laboratory (NREL) from the US for the techno-economic analysis of renewable energy projects (Figure 4). It supports the modelling of various renewable energy technologies, including PV systems, battery storage, wind power, concentrated solar power, and others. SAM's robust financial models allow users to analyse projects from small-scale residential installations to large utility-scale power plants, providing insights into the technical, economic, and financial feasibility of these systems.

Figure 4. SAM´s steps to modelling and technologies/financial options [58]

In addition to the desktop application, SAM includes a software development kit (SDK), which offers integration tools such as NREL-PySAM **[7]**, a Python package that allows users to access SAM's Simulation Core (SSC) programmatically (It should not be confused with the unrelated "pysam", a python genomics package that shares a similar name on PyPI). NREL-PySAM enables flexible, customizable modelling of renewable energy systems directly in Python, offering a robust set of functions to adjust parameters and simulate scenarios. It is distributed with a BSD 3-Clause license, allowing broad flexible use of the code with suitable author and license attribution.

The input data that a PySAM model needs is based on what desktop SAM itself requires: a valid time series renewable resource file, the core technology design parameters (e.g., PV DC capacity, wind turbine power curve) and economic parameters. The required inputs on each PySAM model can ascend to 500 parameters. The recommended workflow stated in the documentation starts with a prebuilding of the model parameters in the SAM desktop application and importing them into PySAM. There is also the option to create the input parameters from defaults files included with SAM installation. This can be a challenge when aiming to integrate the models in a flexible simulation environment as the projected AVPP, and the approach will likely need to set up several defaults inputs and allowing customisation of a small set of variables.

Internally, PySAM converts the inputs into a chain of Python objects—one per compute module—each exposing grouped inputs, outputs and intermediate state variables. The objects wrap compiled SAM's Simulation Core code accessed via the lower-level PySSC library, so the heavy numerical work (e.g. solar-field optics, battery dispatch optimisation, or stochastic cash-flow analysis) runs at C/C++ speed. Modules share data through the from_existing() constructor, ensuring a common memory space and allowing upstream results (such as hourly electricity generation) to feed downstream grid and finance models; a single execute() call then steps through the chain in the correct order. This modular architecture reproduces SAM's coupling logic, providing parity with desktop runs and supporting advanced hybrid layouts like PV-wind-battery systems.

When executed, a PySAM instance yields a rich set of outputs ranging from high-resolution production time-series and component state variables to aggregated annual metrics (net AC yield, capacity factor, degradation losses) and full financial indicators such as net present value, internal rate of return, levelized cost of energy and pay-back period. All results are accessible through each module's Outputs.

Application and integration within the project

In the context of a possible integration within the AVPP, PySAM could play a pivotal role. PySAM's capabilities allow the simulation of various renewable energy technologies, and it directly provides the modelling of technical and financial aspects, making it an ideal platform for integrating hybrid energy systems within the AVPP. From the technical point, its explicit Python API could be wrapped in asynchronous web workers or micro-services, and its inputs/outputs naturally serialise to JSON for interaction with browser front ends. As it has been mentioned, the main challenge may be the managing of the large input space: maintaining master JSON templates generated in the SAM GUI and patching only the variables that change could be a feasible solution.

HOMER

HOMER is a family of proprietary tools for techno-economic design and optimisation of hybrid microgrids and distributed energy resources (Figure 5). The suite includes HOMER Pro (desktop application) [8], HOMER Front (a web-based environment oriented to screening and customer-facing design workflows) [9], and also the HOMER SaaS API for embedding the HOMER engine inside custom applications [10]. The software originated at the U.S. Department of Energy's National Renewable Energy Laboratory (NREL) as the "Hybrid Optimisation Model for Multiple Energy Resources" [11] and was later commercialised by HOMER Energy; today it is offered by HOMER Energy by UL Solutions as a licensed, closed-source product line.

Homer works with inputs like resource data for RES sources, system parameters (AC/DC coupling, converter topology, interconnection limits and losses), storage parameters (such as battery technology, power, energy sizing and degradation assumptions), and economic assumptions (for example, market price strips for the targeted energy and capacity products).

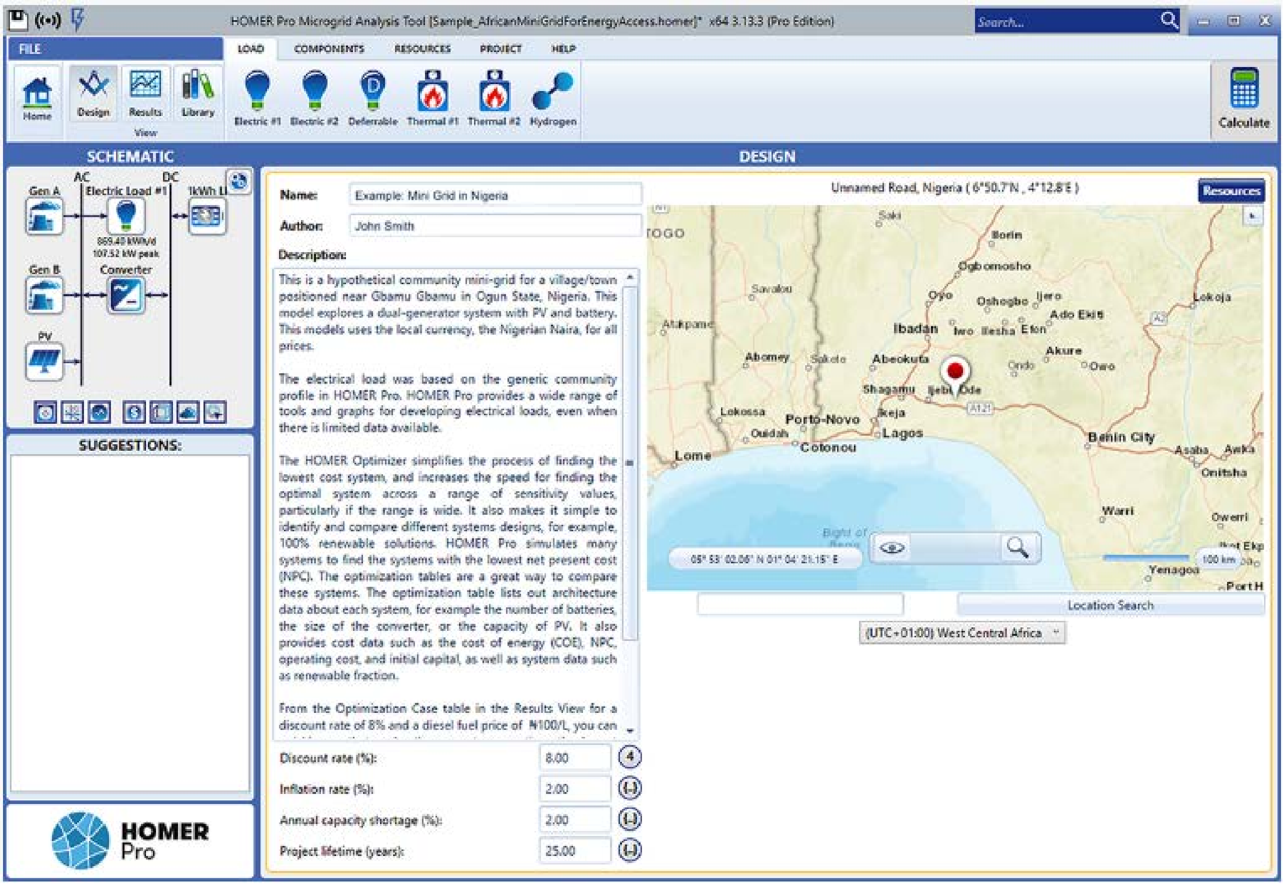

Figure 5. Example of HOMER simulation [59]

Internally, HOMER operates as a chronological simulation with an energy-balance calculation each time step that dispatches generators and storage to meet demand subject to user-defined constraints, then aggregates costs over the project lifetime. An optimisation with an algorithm that searches for the least-cost configurations. The software also supports multi-parameter sensitivity analysis to expose how uncertainty in exogenous variables shifts the optimal design. The model outputs include ranked feasible configurations by economic metrics, along with dispatch profiles, state-of-charge trajectories, curtailed energy, market transactions and cash-flow summaries.

Application and integration within the project

Regarding integration into the AVPP web tool, the most practical pathway would be the use of HOMER SaaS API, which exposes capabilities of the HOMER engine for use in custom web applications. In architectural terms, a Python web backend can orchestrate model creation, execution and retrieval of results via the API, translating between the project’s internal data schema and HOMER’s inputs and outputs. Strengths of this approach are faster time-to-integration and a maintained solver; but the constraints include dependence on commercial licensing, lack of source-level transparency, and possible throughput limits or data-sovereignty considerations inherent to SaaS.

3.2 Agricultural Modelling

A variety of software libraries exist for simulating agricultural systems, ranging from open-source frameworks to proprietary tools. They differ in complexity, target users, required inputs, and provided outputs.

As part of the present project, the models should be ideally suitable for a convenient integration in a web application developed with Python as a basis, and in the best scenario, with potential to be integrated into an agrivoltaic scenario. A comprehensive review of currently available tools for the simulation of agronomic systems is conducted, with the objective of selecting the most suitable solution for the integrated assessment of both agricultural and RES productivity in the future development of the AVPP.

Below are described some of the available agriculture simulation tools, highlighting their licensing, complexity, typical users, inputs/outputs, Python integration and SaaS/cloud readiness.

3.2.1 *Agricultural Production Systems Simulator (APSIM)*

APSIM [12], [13], [14] is a highly modular agronomic ecosystem modelling framework widely used in research and agronomy. It can simulate a broad range of cropping systems with detail, with recent developments [15] extending APSIM to perennials like grapevines, making it relevant for vineyards. Its complexity is high, intended for advanced users who need to explore management strategies or climate impacts on productivity (Figure 6). The APSIM source is publicly available but under a custom license. It is free for non-commercial use under a *General Use License*, but commercial use requires a separate agreement. The APSIM Initiative retains intellectual property rights to the model, so while users can view/modify code, any improvements must be shared back with the developers. In summary, APSIM is not OSI-licensed open-source (due to the commercial use restriction), but it is community-developed and accessible for research purposes.

Some of the multiple inputs which APSIM requires are daily weather data (rainfall, temperature, radiation), soil profile characteristics, crop genetic parameters, and management actions (planting, fertilizer, irrigation schedules). Its outputs include time-series of crop development (phenology dates), biomass growth, final yield, and resource use (e.g. soil moisture and nitrogen dynamics). APSIM simulates water use and soil water content in detail and can track nitrogen leaching and other environmental indicators. It does not directly output surface albedo, which could be useful in a bifacial PV integration. However, it does model canopy cover/LAI which could be used to estimate albedo externally. Economic results are not a core output, but APSIM can be linked to external economic analysis.

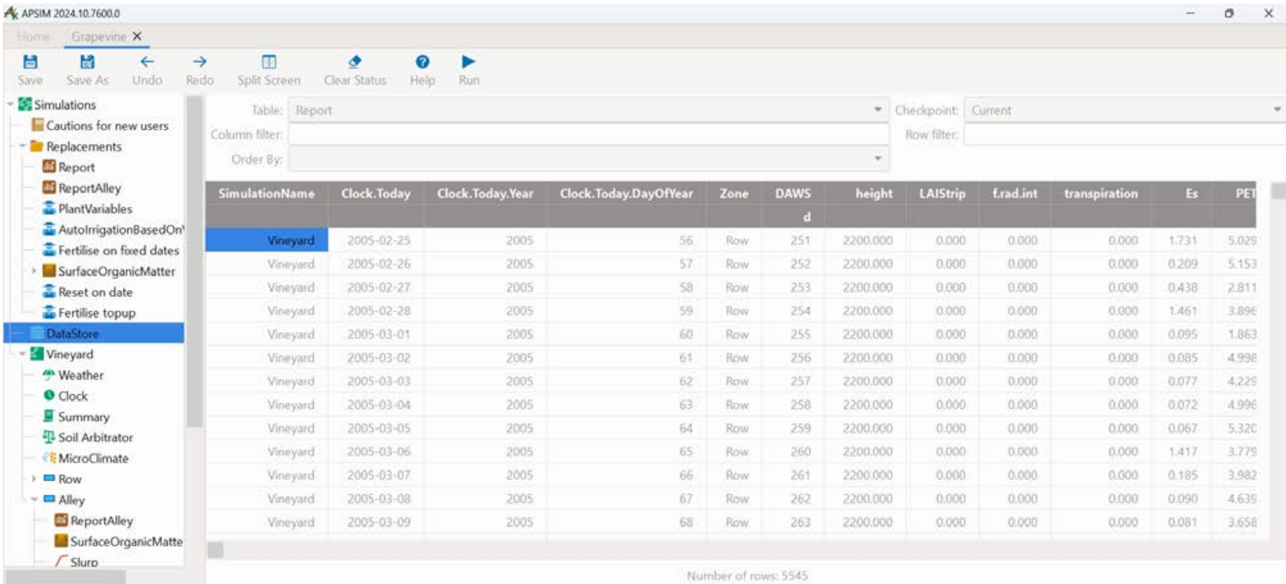

Figure 6. Screenshot of APSIM Next Generation software

Application and integration within the project

Its ability to output detailed crop growth and soil data means it can be coupled with renewable energy models (e.g. using crop height/LAI to inform solar shading or water use to inform energy for irrigation), though custom coupling is needed. APSIM Next Gen is implemented in C#/.NET and runs on Windows, Linux, and macOS. While it does not have an official Python API, it supports automation via scripting or wrappers. For instance, an R package exists to run APSIM-X simulations. A hypothetical integration could be achieved in Python by calling APSIM's command-line or through .NET interoperability, but it would require advance development. Also, its more restrictive license could pose a barrier to possible implementation within the project. Although its license is open in nature, it is relatively more restrictive, which could represent an obstacle to its implementation within the project.

3.2.2 *Decision Support System for Agrotechnology Transfer (DSSAT)*

The Decision Support System for Agrotechnology Transfer (DSSAT) **[16], [17], [18]** is a long-established crop simulation suite that includes models for over 40 types of crops. It is designed for researchers, educators, consultants, and policymakers in agriculture (Figure 7). It can be applied for scenarios like climate change impacts, precision farming, and food security analysis. DSSAT's Cropping System Model code is open and hosted on GitHub [19] under a BSD 3-Clause license. This permissive license allows use and modification (including commercial use) as long as attribution is given.

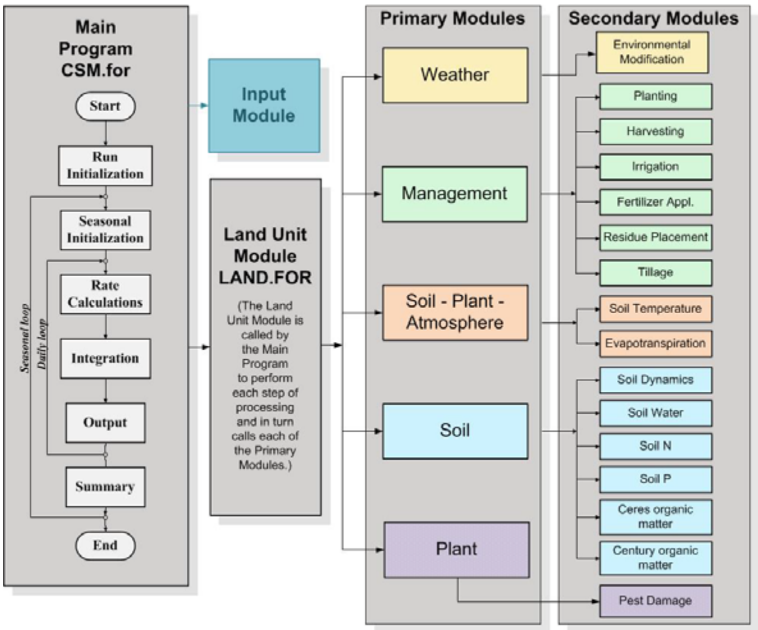

Figure 7. DSSAT Cropping System Model schematic [60]

The tool can be run via CLI by calling the 'subprocess' module. For this purpose, third party adaptations of DSSAT exists, as pyDSSAT [20], or DSSATools [21]. These Python wrappers libraries enable users to develop scripts in python for running simulations within the DSSAT modelling framework. DSSATools uses the GPL-3.0 license, allowing the software to be used, modified, and distributed, but provided that any derivative work, modification, or redistribution is also released under the same GPL-3.0 license.

DSSAT requires daily weather data (including solar radiation or sunshine, temperature, rainfall), soil profile data (texture, depth, initial moisture/nitrogen), crop genotype coefficients (cultivar-specific parameters), and management practices (planting date, fertilizer and irrigation events). Its models simulate crop growth, development stages, and yield as functions of soil–plant–atmosphere dynamics.

The tool can output final yields (grain, biomass), detailed daily crop growth metrics (LAI, biomass accumulation), soil water content and uptake, and soil nutrient status are typical outputs. Water consumption (evapotranspiration, runoff, percolation) and nitrogen leaching are modelled, supporting environmental and sustainability analyses. DSSAT does not report canopy albedo explicitly, focusing instead on biophysical variables; however, it simulates radiation interception by the canopy. Economics results are not inherently provided. Some tools like DSSAT's seasonal analysis can optimise management economically, but any cost/profit calculations are done outside the core crop model.

Application and integration within the project

As mentioned, third-party Python interfaces exist to facilitate running simulations from Python. Although they rely on individuals rather than being backed by a full organisation, the incorporation of DSSAT into Python-based applications or cloud services could be feasible. The wide variety of crops and the complexity of inputs and outputs increase its potential applicability across a broad range of scenarios. However, it would still be necessary to narrow down the inputs by compiling a set of default cases. DSSAT's permissive license is an advantage, however, in the case of DSSATools, the requirement to maintain GPL-3.0 licensing for derivative works would make its integration into the AVPP incompatible and would necessitate the development of custom wrappers.

3.2.3 *AquaCrop (FAO AquaCrop and AquaCrop-OSPy)*

AquaCrop [22], [23], [24] is a crop model developed by the FAO UN to simulate crop yield response to water supply. It focuses on water-limited environments, targeting agronomists, irrigation specialists, and development practitioners concerned with water productivity and food security. The model has been parameterised for a wide variety of crops and is known for requiring fewer crop-specific inputs by using canopy cover growth curves and water productivity parameters.

The official AquaCrop model is distributed in two main formats: a standard version with a graphical user interface (GUI), and a stand-alone version designed for automated or large-scale batch simulations. These two formats share the same computational core. Since version 7.2 [25], the computational core, rewritten in Fortran, is released on GitHub under the permissive BSD-3-Clause licence, while the accompanying graphical interface and databases remain under FAO copyright (redistribution for commercial purposes still requires FAO clearance). An earlier, fully open-source re-engineering appeared in August 2016 as AquaCrop-OS [26], coded in MATLAB/Octave by a consortium led by the University of Manchester together with the Water for Food Global Institute, FAO, and Imperial College London. Later, building directly on that code base, the developers produced AquaCrop-OSPy [27], a pure-Python port distributed through PyPI, which keeps the same Apache 2.0 license. AquaCrop-OSPy provides identical science routines to AquaCrop, written in pure Python, offering even wider freedom for reuse and more flexibility in its implementation. As the official documentation indicates: "[…] can be easily integrated with the latest machine learning, web development, statistical and data analysis libraries, as well as any other Python code base".

AquaCrop key inputs include daily weather data (at least rainfall, temperature, and reference evapotranspiration or sunshine to compute ET_0), soil profile characteristics (layer depth, texture, readily available water), crop parameters (base temperature, growth duration, canopy expansion rate, water productivity, etc.), and field management (planting date, irrigation schedule, fertility level) (Figure 8).

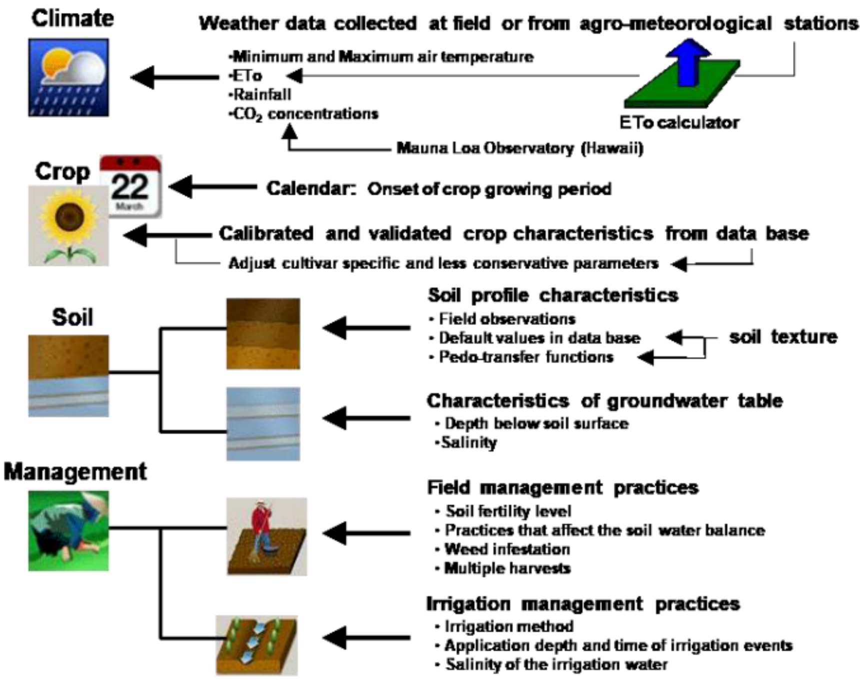

Figure 8. AquaCrop Inputs [28]

AquaCrop simulates crop development (green canopy cover over time), final yield and biomass, soil water content by depth, and season-long water consumption (evapotranspiration, transpiration vs. evaporation) under given water management. It explicitly calculates crop water productivity and can report water-use efficiency. Economic results are not included in the core model, as AquaCrop sticks to biophysical outcomes.

Application and integration within the project

Integration into a Python-centric, web-deployed energy systems platform is technically straightforward. Thanks to the AquaCrop-OSPy python library and its cloud-friendly characteristics, it could be deployed in web applications or containers. The official FAO AquaCrop software also provides a command-line mode, which could be used on servers if needed, though it is a .NET application. This model is well-suited to coupling with water-related energy models (for example, calculating the energy needed for irrigation pumping based on AquaCrop's water use output). In a possible coupling with solar energy models, its prediction of canopy cover could be used to estimate solar panel shading in an agrivoltaic scenario, although it would require advanced custom integration.

3.2.4 *Python Crop Simulation Environment (PCSE)*

PCSE (Python Crop Simulation Environment) [29], [30] is an open-source Python package that provides a framework for building and running crop simulation models. It was developed at Wageningen University to modernise well-known crop models like WOFOST (World Food Studies), LINGRA and LINTUL, which were originally in Fortran. PCSE is targeted at researchers and developers who want to incorporate crop models into their own applications or workflows. It emphasises flexibility and integration (e.g. working with databases, web apps, containers) rather than being a turnkey GUI tool. PCSE is open source under the European Union Public License (EUPL). This is a copyleft license that ensures any derivative code of PCSE remains open source, while permitting commercial use subject to the EUPL requirements [31]. In a SaaS or open-to-users context, the license requires preservation of PCSE's copyright and license notices, attribution to the author, and public availability of its source code. These obligations apply to PCSE itself and to any modifications of it, and do not extend to independent components of the AVPP.

To run a simulation, PCSE expects three categories of input. Parameter sets describe crop physiology, soil hydrology and site-specific settings. Weather time series data, with daily resolution, such as radiation, temperature, wind and precipitation feed its time-stepping engine, while an agro management YAML file records the farming calendar (sowing, harvest, irrigation, etc.). The framework's ParameterProvider object bundles these disparate sources so that a single typed interface is passed to the model instance. NASA-POWER and other built-in data providers make it possible to fetch global weather data on-the-fly, illustrating PCSE's flexibility for large-scale scenario studies.

Internally, simulations are executed by an engine that advances state variables with a fixed daily time step. Each physiological or biophysical process is encapsulated in a module that exposes clearly defined rate and state variables; the engine orchestrates their initialisation, rate calculations and state updates. Typical outputs include crop development stage, leaf-area index, above-ground biomass, root-zone soil moisture and marketable yield. Such rich diagnostic information makes the library attractive for coupling with resource-use or climate-impact assessments (Figure 9).

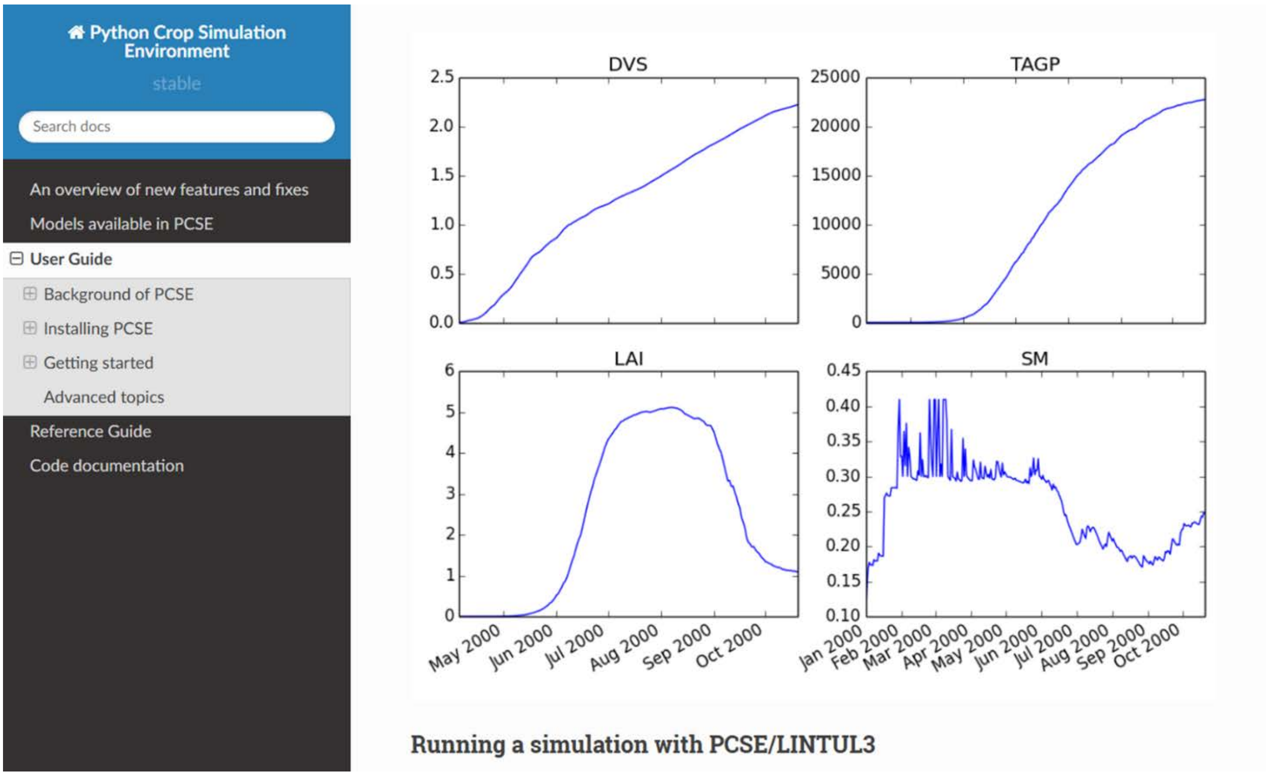

Figure 9. PCSE. Crop Simulation example [61]

Application and integration within the project

Being natively written in Python makes it straightforward to integrate into larger Python projects or web services. It is well-documented and can read/write JSON or CSV data for inputs/outputs, facilitating its use in a SaaS platform that, for instance, collects input data from users and returns simulation results. Because PCSE can be run stepwise and customised, it is feasible to couple it with renewable energy models. For example, in an agrivoltaic scenario, an approach could be adjusting PCSE's intercepted radiation input based on solar panel shading, or conversely using PCSE's simulated canopy cover to inform a solar energy model of the ground albedo or shading factor. This flexibility makes PCSE a possible option when tight coupling between crop and energy simulations is needed. One potential drawback to consider is the fixed daily resolution, which is usually adequate for long-term biomass supply modelling but may be coarse for sub-daily energy-water coupling studies. The main potential barrier to the integration of PCSE lies in possible differences in the interpretation of the EUPL regarding what constitutes a “derivative work” in a tightly integrated environment, which could introduce legal uncertainty. As a precaution, and to minimise potential licensing constraints, alternative agricultural modelling tools released under more permissive licenses may be considered.

3.2.5 *Python Agrivoltaic Simulation Environment (PASE)*

PASE (Python Agrivoltaic Simulation Environment) is an academic simulation tool developed by Roxane Bruhwyler at the University of Liège’s, as a PhD thesis [32], [33]. It specifically designed to jointly model the agricultural, solar, and geometric components of agrivoltaic systems and quantify both PV production and crop performance in shared land-use schemes (Figure 10). Its code is distributed under the MIT license, allowing for flexible use and sub-licensing of the tool. At the time of writing this report, the code is hosted and provided in a Gitlab repository [34], indicating its installation in a conda virtual environment directly from source code, and it’s not yet distributed in conda-forge channels, nor on PyPI.

The simulation flow in PASE follows a modular structure that begins with the definition of input files in YAML format. These files contain detailed site-specific information, including geographic location, daily and hourly meteorological data (solar radiation, temperature, relative humidity, wind, precipitation), structural characteristics of the PV modules (height, tilt, spacing, orientation, technology type, transparency factor), physiological properties of the crop (photosynthetic efficiency, harvest index, water use coefficients), and soil parameters (soil texture, field capacity, wilting point, rooting depth).

Internally, the workflow is split into three loosely coupled modules. Solar irradiance on every relevant surface is solved with a 3-D ray-tracing kernel, those radiative fields are passed to a crop-growth engine, and electrical conversion is handled by a thin wrapper around *pvlib*. PASE generates a precise three-dimensional geometry of the PV system over the crop area. Through ray tracing algorithms, the model simulates the solar path throughout the year, projecting hourly shadows with high spatial resolution. This spatial mesh is used to calculate direct, diffuse, and reflected irradiance received at each ground cell, capturing with the actual light distribution under various panel configurations. The resulting daily radiation maps are combined with meteorological conditions to feed PASE’s internal agronomic model. This component simulates the crop's development over its vegetative cycle, computing key variables such as evapotranspiration, Leaf Area Index, accumulated biomass, and fresh yield.

Among PASE’s main outputs are not only the annual balances of energy production and agricultural yield but also a wide range of high-value results, such as: 2D spatial maps of agronomic performance, time series of crop development, daily radiation distribution patterns, soil moisture dynamics, and agrivoltaic productivity estimates under comparative scenarios. This richness of output enables sensitivity analyses, structural optimisation, and sustainability assessments of integrated systems.

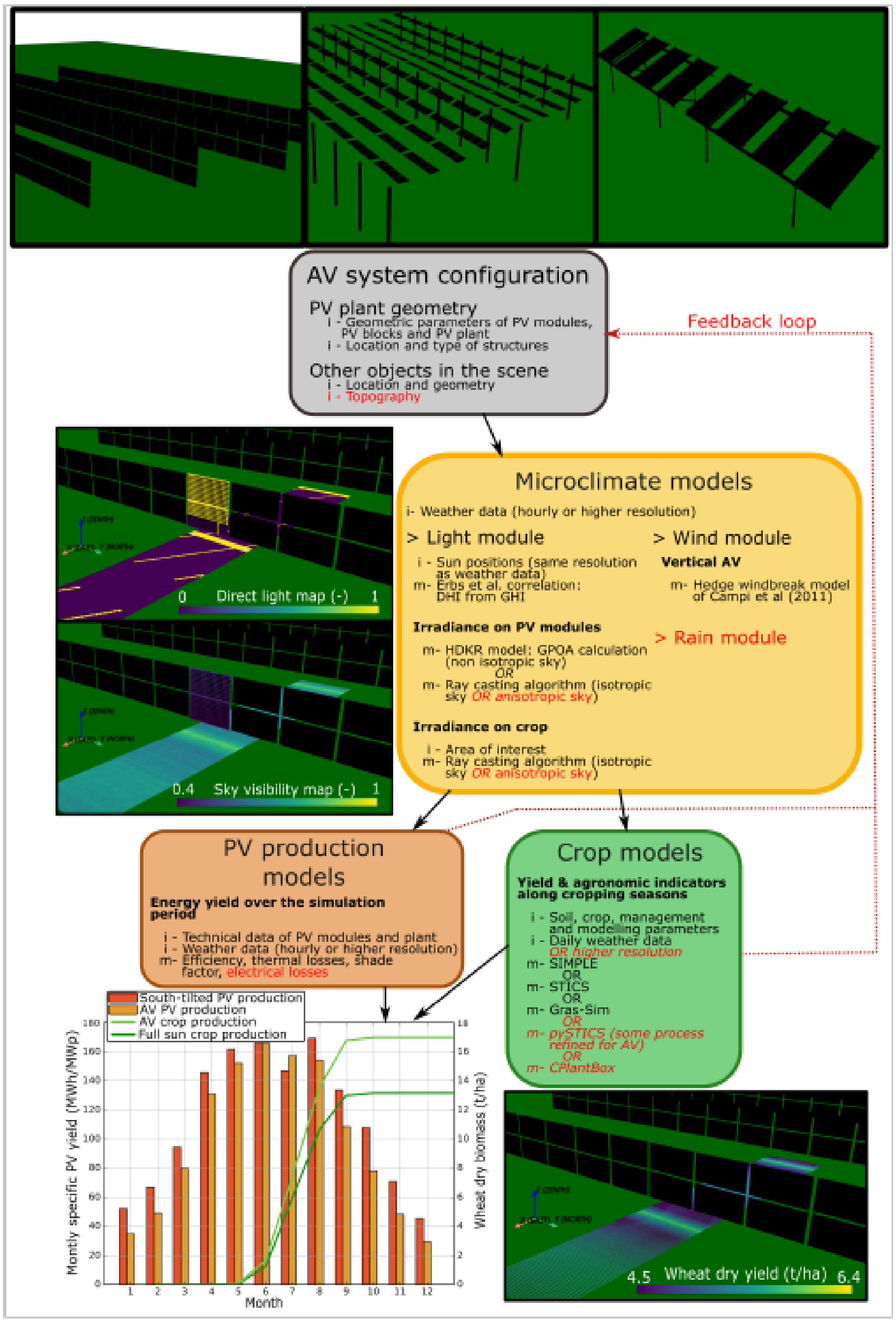

Figure 10. Organigram of PASE 1.0: main modules, links between module, (i-) inputs, (m-) models available and future developments in red [32]

Application and integration within the project

From an integration standpoint, PASE offers three immediate advantages: it is written entirely in Python, so it can live in the same interpreter as a broader analytics stack; it couples PV and crop-growth physics in a single run, eliminating the need to juggle separate tools; and its MIT licence permits redistribution, modification and commercial use without legal friction. Nevertheless, the current distribution model still imposes some practical hurdles. The source code is available only through the public GitLab repository and must be cloned and installed into a Conda environment from source; no wheels are published on PyPI and no recipe exists on the conda-forge channel. Also, its internal modules are not exposed as a formal API and its file paths are hardcoded rather than package-relative, so it cannot yet be pulled in cleanly as a dependency via requirements.txt, environment.yml or pyproject.toml. One workaround would be to vendor the code, i.e., copy it into the host project and patch the packaging, but that would sever automatic tracking of upstream changes unless a synchronisation workflow is maintained. Until the maintainers package PASE for the Python Package Index or conda-forge and refactor the I/O layer to be path-agnostic, a potential integration should be budgeted in time, for both code refactoring and local packaging of the repository.

3.3 Demand modelling

There are different approaches and tools available to generate demand profiles for various working scenarios. For the AVPP environment, time series of consumption data are required across multiple sectors (residential, industrial, agricultural, etc.). These demand profiles are essential to realistically simulate and evaluate the behaviour of the aggregated virtual power plant under different operating conditions.

Several relevant open-source options have been identified for generating such demand profiles, each offering different levels of detail and sector coverage. Among the most prominent are as follows.

3.3.1 *Open Energy Modelling Framework (Oemof)*

Oemof (Open Energy Modelling Framework) is a modular framework that includes demandlib, a library for generating synthetic electrical and thermal demand profiles across sectors (e.g., residential, commercial) (Figure 11).

Demandlib is an open-source Python library [35] for generating synthetic energy demand profiles based on standardised German methodologies, primarily BDEW (Bundesverband der Energie- und Wasserwirtschaft) and VDI 4655. It can create electrical, thermal (heating and domestic hot water), and industrial demand profiles with configurable temporal resolution (15-minute or hourly). Demandlib allows users to generate normalised standard load profiles and scale them to any specified annual demand, making it highly adaptable for scenario-based energy system modelling.

The library requires inputs such as the simulation year, annual energy demand per sector or building, selected sector profile (e.g., H0, G1-G6, L0-L2), and optional holiday calendars. For heat and residential profiles, it can use weather data (TRY or user-provided) and building parameters, including type, occupancy, and construction class. Industrial profiles can be customised with working hours and day/night scaling factors. Demandlib is released under the MIT License, making it freely usable, modifiable, and suitable for integration into energy system modelling workflows.

Figure 11. Oemof framework & Demandlib python library [62]

Application and integration within the project

For the HarvREStproject, Demandlib could be used to provide hourly or sub-hourly demand time series for various sectors (residential, commercial, agricultural, and industrial). Its ability to scale standardised profiles to specific annual consumption values allows realistic scenario definition tailored to the AVPP module requirements. By integrating climatic data and localised calendars, Demandlib can also simulate heat demand dynamics relevant to combined energy systems. This makes it a flexible, open-source solution well-suited for generating diverse demand datasets to test aggregation strategies, energy management algorithms, and sector-coupling scenarios within AVPP.

3.3.2 *LoadProfileGenerator (LPG)*

LoadProfileGenerator (LPG) [36],[37] is an agent-based simulation tool designed to generate high-resolution synthetic residential load profiles for electricity and domestic hot water consumption. It uses a desire-driven behavioural model to simulate detailed daily activities of household residents and their impact on energy use, producing profiles with a temporal resolution of 1 minute. The tool includes predefined German households, customizable household configurations, and behavioural traits (Figure 12).

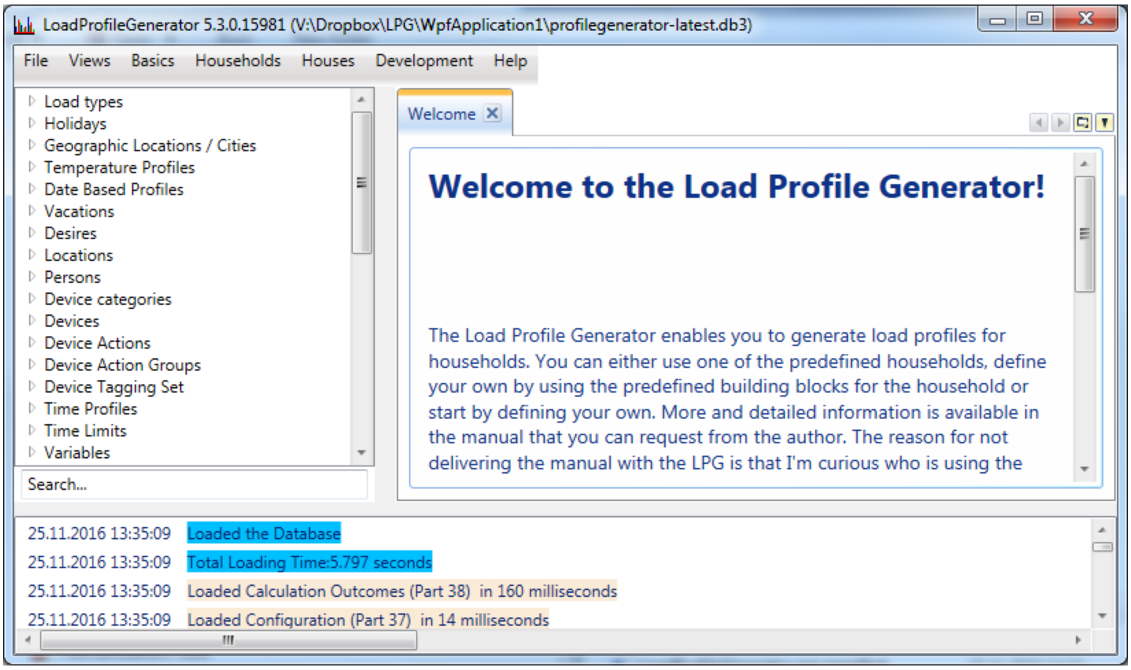

Figure 12. Load Profile Generator [63]

Originally developed with a graphical user interface for Windows, LPG has recently been released as open-source under the MIT license and now includes a Python wrapper (pylpg), enabling automation and integration into Python-based workflows.

These are the main features of this tool:

- The LPG is a modelling tool for residential energy consumption
- It performs a full behaviour simulation of the people in a household and uses that to generate load curves.
- It has various German households predefined.
- It generates CSV files for the import into other simulation software.
- Additionally, it generates a ton of different reports to help with modelling.

- The focus is the individual household. It is possible to evaluate larger populations, but this will require significant work

Application and integration within the project

LPG provides synthetic, high-resolution (1-min) residential load profiles that could be used as demand inputs for the AVPP module, enabling realistic modelling of household consumption behaviour in virtual power plant simulations. Its Python wrapper, pylpg, allows seamless automation and integration into AVPP's Python backend, removing the need for manual GUI operations and enabling direct interaction through scripts. As an open-source tool (MIT license), LPG offers validated predefined German households as credible baselines while remaining fully customizable. Its CSV outputs could be easily imported into AVPP for testing aggregation strategies, energy management scenarios, and demand flexibility, making it ideal for fine-grained, automated control simulations.

However, to date, this tool is limited to residential demand profiles, while for the project it would be desirable to include additional demand types such as industrial facilities, farms, and other non-residential sectors.

3.4 Meteorological data collection

As input for the different models described in this deliverable, it is common to use meteorological data, as well as simplified representations in the form of typical years. These are hourly time series of one-year duration (8,760 hours) that capture the typical or average behaviour representative of a long-term period (20–30 years). To construct these time series, it is necessary to gather variables such as wind speed, solar radiation, temperature, pressure, among others, which can be obtained from various sources. This section highlights some of the most common and accessible options.

3.4.1 *Copernicus data*

Copernicus, It is a project coordinated and managed by the European Commission, aimed at achieving a complete, continuous, and autonomous high-quality Earth observation capability, with results freely accessible to the scientific community, public authorities, and individuals [38]. Among the many available datasets included in this programme (e.g., climate monitoring, atmosphere, land, and marine data), the most relevant for this project are the ERA5 reanalysis datasets [39], which provide hourly meteorological data with global coverage and a resolution suitable for energy system modelling. ERA5 includes key variables such as wind speed, solar radiation, temperature, and pressure, making it a widely used source for renewable energy simulations (Figure 13).

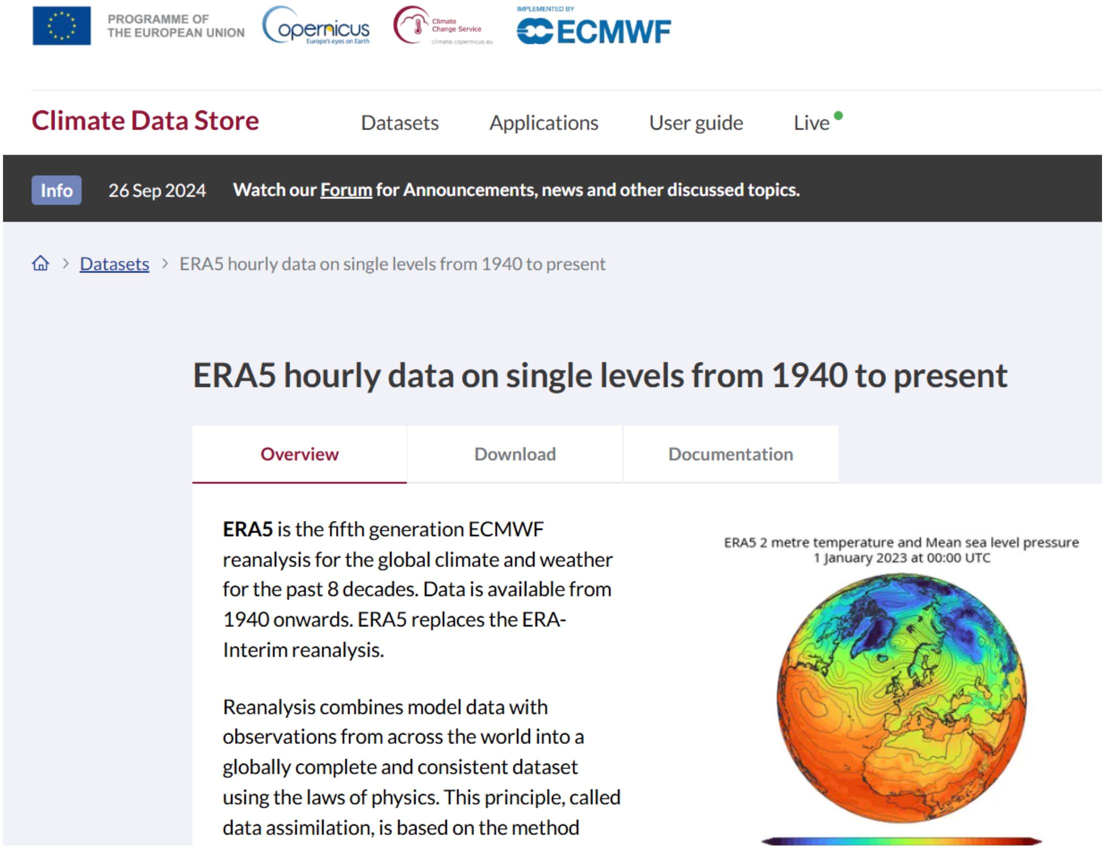

Figure 13. ERA5 reanalysis datasets [39]

Application and integration within the project

The Copernicus ERA5 reanalysis dataset is a free and feasible source of hourly meteorological data for integration into AVPP, offering global coverage and suitable variables for energy applications. However, its limited download performance and latency make real-time integration challenging, so its use is best suited for batch processing, validation, or non-immediate analysis scenarios.

3.4.2 *Openmeteo*

Open-Meteo [40] is a free API service that provides global weather data with hourly resolution, suitable for renewable energy modelling. It offers both forecasts and historical data for key meteorological variables, with free access (up to 10,000 daily API calls) requiring no API key or registration (Figure 14). For higher request volumes or commercial applications, a subscription plan is available.

The platform is fully open-source, with its codebase available on GitHub under the AGPLv3 license, allowing users to explore, modify, and deploy their own instances. Its data is licensed under CC BY 4.0, permitting sharing and adaptation, including for commercial use with proper attribution.

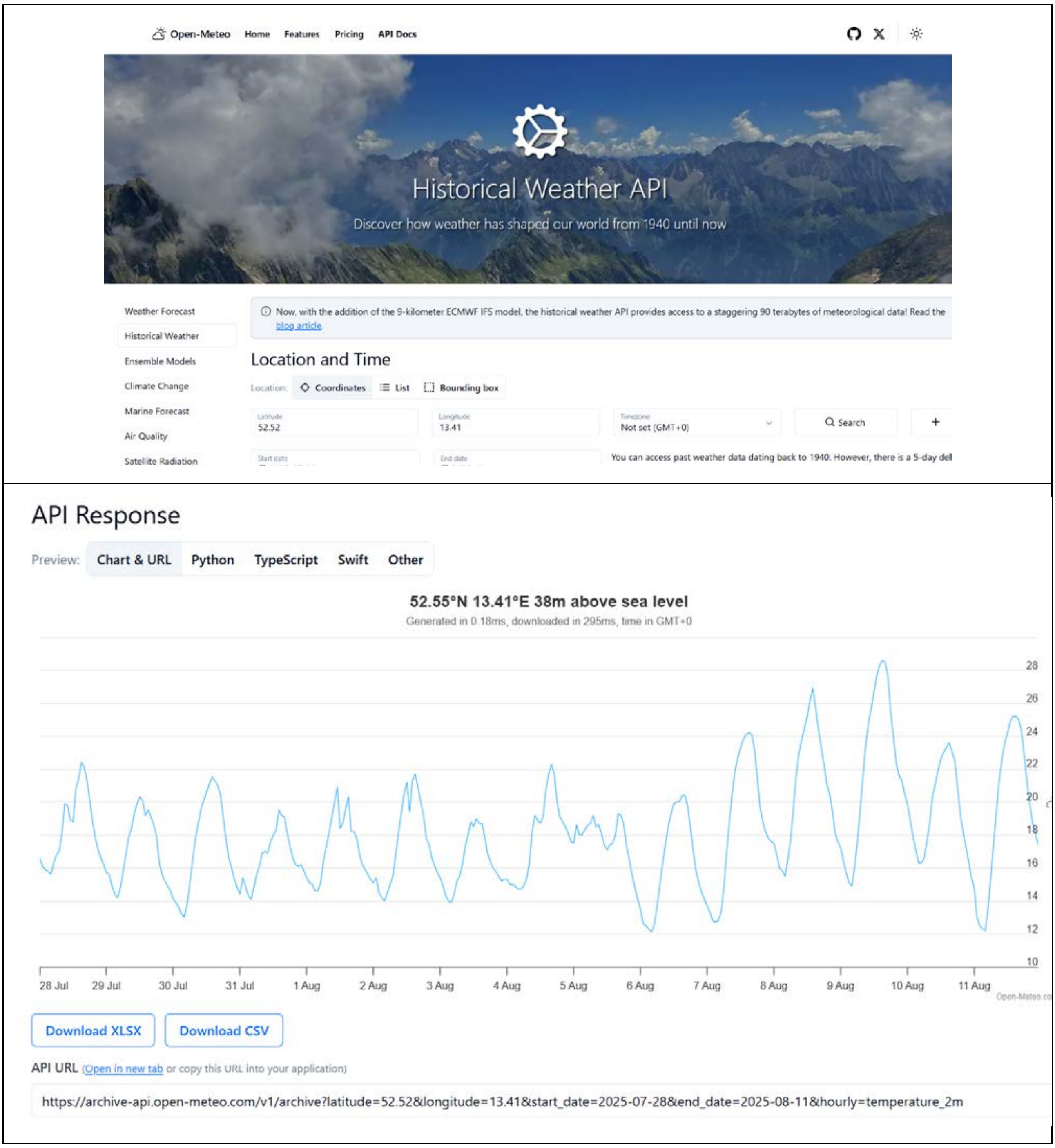

Figure 14. Open meteo API [64]

Application and integration within the project

Open-Meteo's open-source API is particularly well-suited for integration into the AVPP due to its programmatic access, permissive licensing, and compatibility with Python-based workflows. Its ability to provide hourly weather data (including wind speed, solar radiation, and temperature) without requiring registration or API keys makes it ideal for automated data retrieval and scenario modelling.

The AGPLv3-licensed codebase and CC BY 4.0 data license ensure transparency and allow for customisation or deployment of a dedicated API instance if higher performance or control is required. Additionally, Open-Meteo offers paid API plans for intensive or commercial use, providing extended access and support beyond the free tier limit of 10,000 daily API calls. This flexibility aligns with the AVPP's need for reproducible, scalable meteorological data pipelines that support renewable generation modules and enable frequent updates for long-term simulations and sensitivity analyses.

3.4.3 *Global Wind Atlas (GWA)*

The Global Wind Atlas (GWA) [41] is a free, web-based platform developed to support policymakers, planners, and investors in identifying high-wind areas worldwide for preliminary wind power assessments. It provides high-resolution wind resource mapping (250 m grid) both onshore and offshore (up to 200 nautical miles) and allows users to visualise wind speed and power density at multiple heights (10–200 m) (Figure 15). The tool enables online queries and offers freely downloadable datasets, including GIS-ready maps, wind resource statistics, and energy yield calculations based on custom turbine power curves. GWA also provides temporal analysis capabilities to assess wind variability by year, month, or hour, and includes complementary data to explore synergies with solar resources.

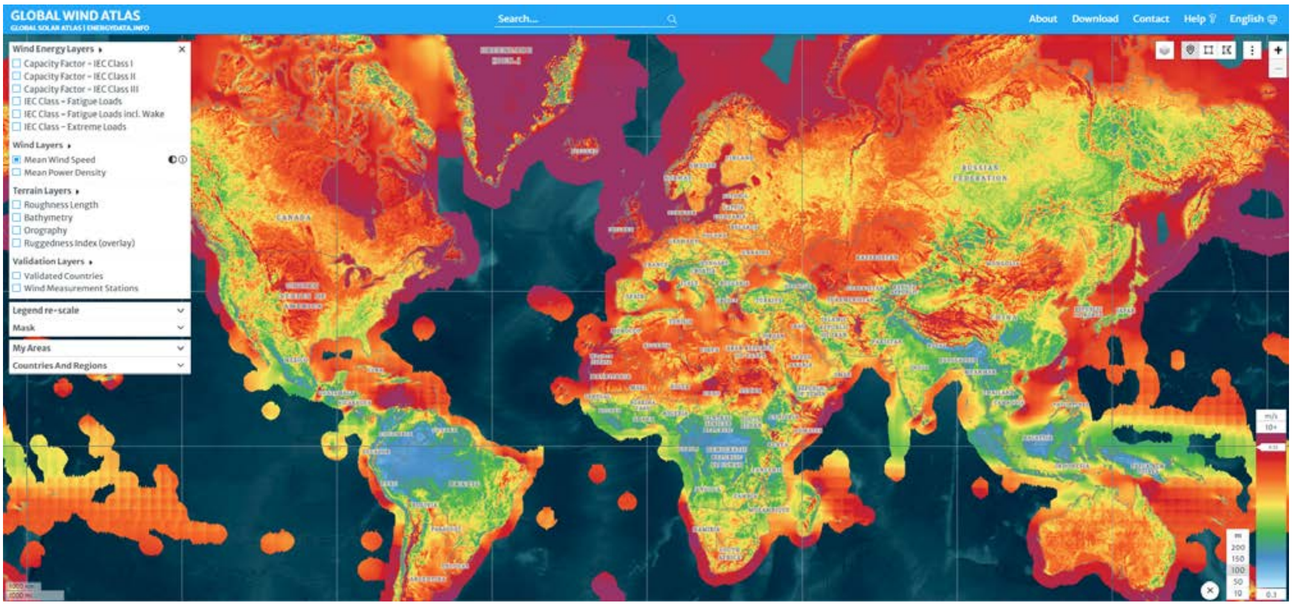

Figure 15. Global Wind Atlas – Mean wind speed at 100 m [65]

GWA is built on advanced downscaling methods that combine ERA5 reanalysis data, mesoscale modelling (WRF), and microscale modelling (PyWAsP) to generate accurate localised wind resource data. It incorporates detailed terrain, roughness, and land-use information, validated against measurement campaigns funded by ESMAP (Energy Sector Management Assistance Program). The tool is publicly accessible without cost or registration and is intended for large-scale resource assessment, energy planning, and integration studies—not for detailed wind farm siting. Its open-access nature, backed by the World Bank and DTU, ensures transparency in methodology and promotes equitable access to high-quality wind resource data globally

Application and integration within the project

The Global Wind Atlas offers high-resolution average wind speed data, but it does not provide hourly time series. While it delivers more accurate spatial representations of wind resources compared to reanalysis datasets, it lacks temporal detail.

A recommended approach is to use Global Wind Atlas data for spatial correction or calibration of the hourly wind speed profiles obtained from ERA5, combining the strengths of both sources.

3.4.4 *Typical meteorological year (TMY)*

Typical Meteorological Year (TMY) datasets provide a synthetic one-year sequence of hourly meteorological data representative of long-term climatic conditions. Initiatives such as PVGIS [42] offer TMY datasets primarily focused on solar resource modelling (Figure 16). However, their direct applicability to wind energy is more limited. This is because the tool is primarily oriented toward solar data, giving it higher priority over wind data. Furthermore, wind data require corrections to hub height (among other adjustments), and their greater variability and sensitivity to local conditions compared to solar radiation necessitate the combination of additional, more precise sources to ensure accurate wind resource representation.

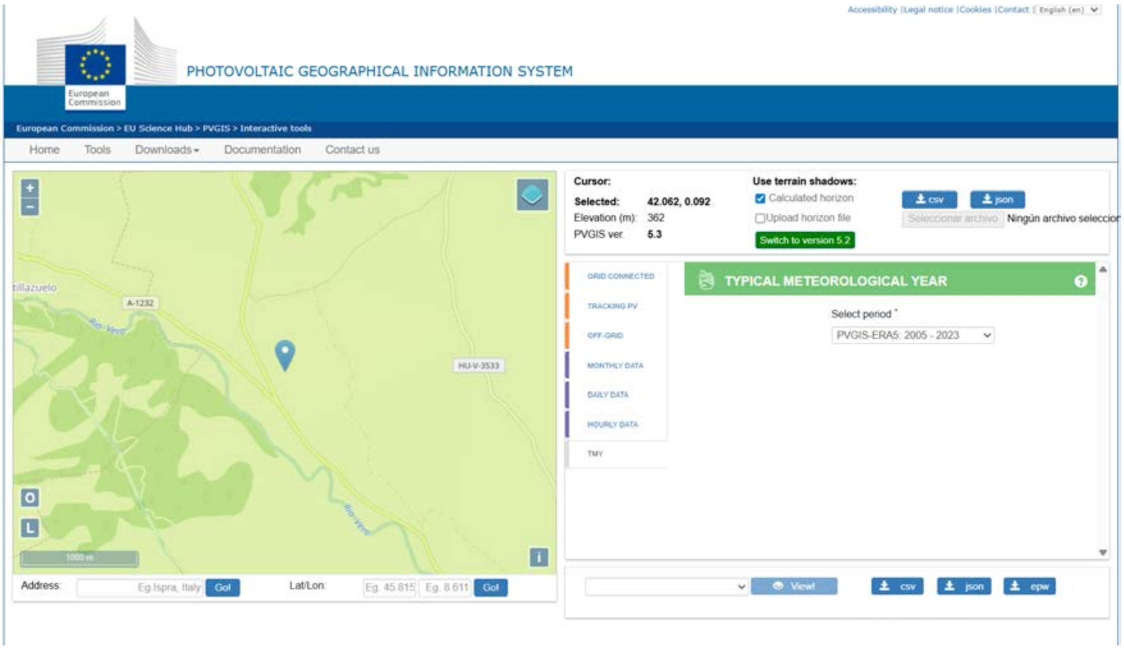

Figure 16. PVGIS tool [66]

Application and integration within the project

For comprehensive renewable energy simulations within AVPP, TMY data can complement reanalysis datasets, particularly for hybrid scenarios where both wind and solar energy need to be modelled. In this context, it is likely necessary to develop a methodology for generating a reference year from long-term datasets obtained from sources such as ERA5 within the framework of the HarvRESt project.

3.5 Other tools

3.5.1 *Biogas modelling*

To integrate into the AVPP tool ideally a simplified tool for dimensioning the generation potential from different substrates is needed. It would be expected to be able to choose the type of substrate, apply digester efficiency, include methane content and calculate useful energy (kWh or MWh), include impact of temperature, Ph, digester residence time. To explore these requirements, some tools have been studied.

For this purpose, identifying accessible and functional tools for estimating biogas production, a systematic web search was conducted for online calculators. The selection focused on free tools, with immediate use (i.e., those providing real-time results without the need for registration or download), and that offered the possibility of working with different substrates. Priority was given to calculators that allowed mixing multiple substrates; alternatively, those including a wide variety of predefined substrates were considered. As a result, seven tools that met these criteria were identified and selected.

Biogas calculators, like the ones listed below, are designed to quickly and accessibly estimate the biogas production potential from various types of organic substrates. These tools are intended for technicians and professionals in the energy and agro-industrial sectors, as well as for researchers, students, and small-scale producers interested in evaluating the feasibility of anaerobic digestion projects. In general, they allow users to choose from different organic wastes (agricultural, agro-industrial, household, among others) and input data such as quantities or substrate proportions to calculate the estimated volume of biogas generated, its energy content, and, in some cases, emission reductions or economic performance. Their free, online, and user-friendly nature makes them valuable tools for a preliminary assessment of organic waste-to-energy projects.

The objective of this analysis was to identify biogas calculation tools currently available on the market. These tools can serve as references or benchmarks to validate the results obtained during the future development of the AVPP. Given that the ultimate goal is to integrate biogas production estimates into the HarvRESt project's in-house tool, Python libraries will be required to enable direct integration and automation (e.g., PyADM1, BiPMo) or to support the creation of a custom biogas module using configurable conversion factors.

If the tools identified do not support Python-based workflows or API access, their methodologies and calculation principles will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework. This approach ensures that, even when existing tools are limited to web-based interfaces, their underlying models can still inform and enhance the biogas simulation capabilities of HarvRESt.

Biogas Calculator – Renergon

RENERGON International AG is a Swiss company founded in 2010, specialising in the development of sustainable technological solutions for biogas production from organic waste. With solid expertise in engineering and anaerobic digestion processes, RENERGON focuses on transforming waste into renewable energy and valuable products, thereby promoting a circular economy.

As part of its commitment to sustainability and innovation, RENERGON has developed a free web application (https://www.renergon-biogas.com/en/biogas-calculator/) designed to estimate the energy and economic potential of a biogas plant based on anaerobic digestion. This tool allows users to input the amount of substrate and co-substrate (in tons per year) to calculate the estimated output (Figure 17).

Users can choose between three energy options: biogas production, electricity generation, or a comparison of both. Additionally, the tool provides approximate estimates of the amount of organic fertilizer and compost generated, as well as the reduction in CO_2 emissions associated with the process. It also calculates the potential revenue from the sale of gas, electricity, compost, and waste disposal fees, thus offering a comprehensive overview of the economic and environmental impact of a biogas plant.

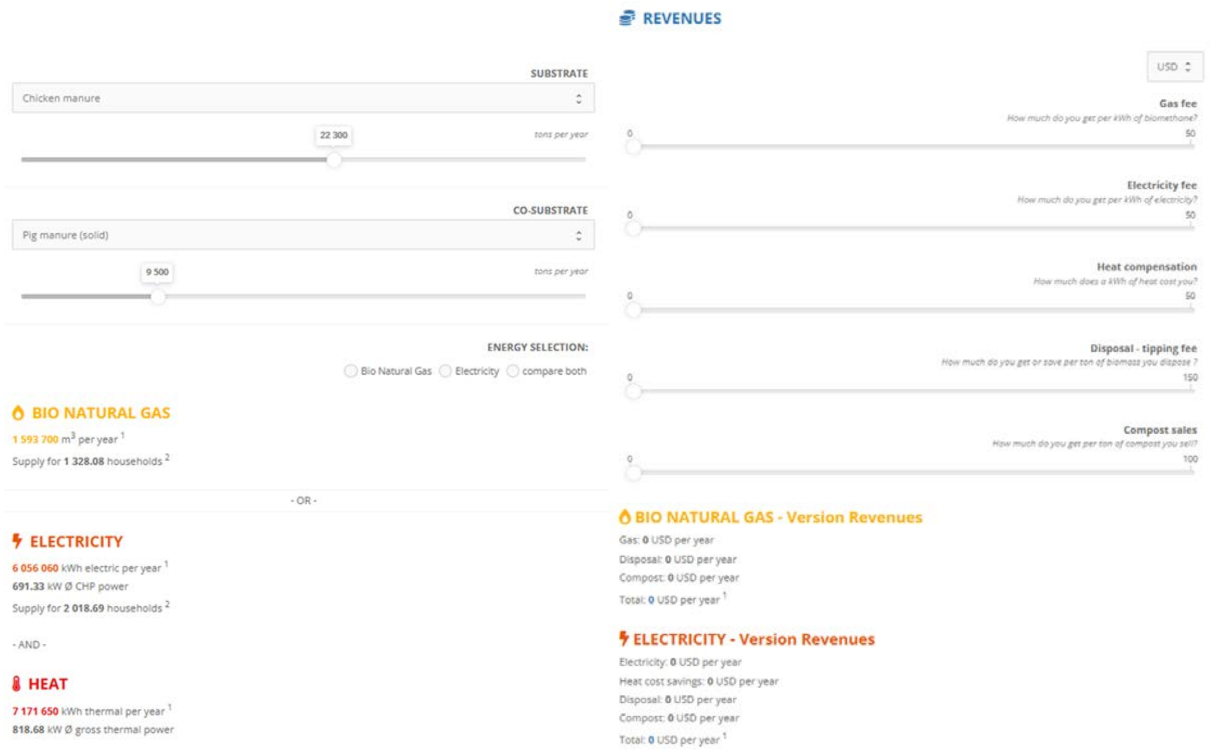

Figure 17. Screenshot of the Biogas Calculator from Renergon

The biogas calculator offers several advantages and disadvantages. Among its advantages, it is free to access and requires no registration, features a user-friendly and intuitive interface, enables fast simulation of results, and includes both environmental and economic aspects. Additionally, it is based on real-world technology. However, it also has certain limitations: it uses standardised average values, does not allow adjustment of all technical or economic parameters, focuses exclusively on dry digestion, and does not replace the need for a feasibility study.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

Biogas Calculations – Biogas World Media Inc.

BiogasWorld Media Inc. is a Canadian company founded in 2015, specialising in connecting product and service providers with project developers in the biogas and renewable natural gas sector. Its goal is to facilitate the development of sustainable projects by promoting clean technologies and innovative solutions in the global energy transition.

BiogasWorld has developed an online free tool (https://biogasworld.com/biogas-calculations/) designed to estimate the technical, economic, and environmental potential of large-scale anaerobic digestion projects. By entering the types and quantities of substrates, the calculator provides approximate estimates of the annual biogas production (in m^3), biogas utilisation, potential revenues from the sale of energy and digestate, as well as estimated investment and operating costs (Figure 18).

It is worth noting that the results are not displayed immediately, but only after the user provides their contact information.

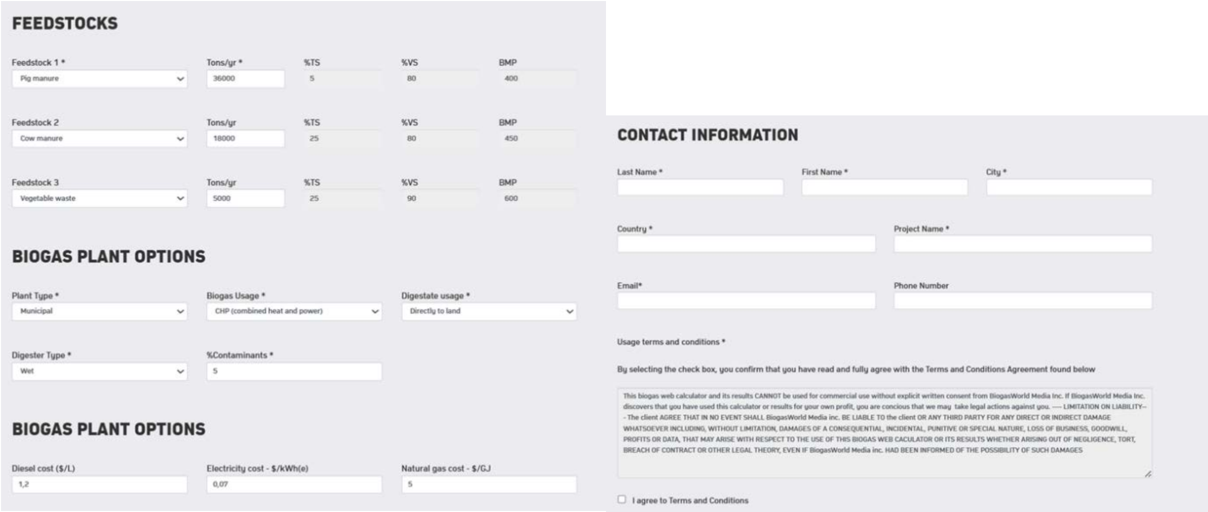

Figure 18. Screenshot of Biogas World Calculations

The BiogasWorld Calculations tool offers several advantages and disadvantages. On the positive side, it is freely accessible online and takes a comprehensive approach by covering technical, economic, and environmental aspects. It also allows users to input specific data for more accurate estimates and is useful for making an initial assessment of a project's feasibility. However, there are some limitations: results cannot be used for commercial purposes without the company's consent, they are approximate and cannot replace a detailed feasibility study, and the tool requires accurate information as well as an understanding of technical concepts.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

Vécycle

Green Island Cécycle ApS is a Danish company founded in 2021, specialising in the development of biogas solutions aimed at the agricultural sector. It is part of Green Island Group ApS, a conglomerate committed to promoting green technologies and sustainable solutions for local communities, with the goal of fostering the transition to a low-carbon economy.

This free online calculator (https://vecycle.dk/calculator/) was specifically designed to help farmers to estimate the potential production of biomethane and the environmental benefits associated with the use of manure and other organic waste.

By entering basic data such as the number and type of animals, the tool provides approximate estimates of potential biomethane production, greenhouse gas emission reductions, avoided emissions, and savings in nitrogen fertilizer (Figure 19). This tool aims to support decision-making and promote the energetic and environmental utilisation of agricultural waste.

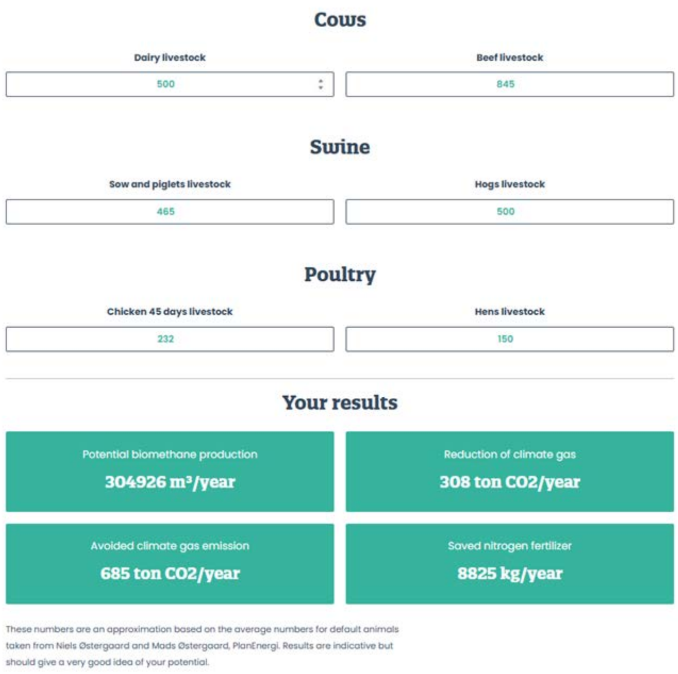

Figure 19. Screenshot of Vécycle

The Vécycle tool offers several advantages, including free and online access, a simple interface, and immediate results. However, it also has some limitations. The estimates it provides are general, and to obtain personalised data, users must provide their contact information. Additionally, it is mainly designed for livestock operations and may not be suitable for other types of organic waste.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

TWIN 4.0

Developed and designed by the company ETE FAROS LTD as part of the TWIN 4.0 project, this online tool (https://twin40.eu/biogas-calculator/) is part of an initiative funded by the European Union through the Erasmus+ program. Its objective is to estimate the potential for biogas and biomethane generation from various types of organic waste.

The tool request the users to input the quantities of different waste types, including pig slurry, cattle manure, poultry manure, fruit waste, food waste, and vinasse (Figure 20). Based on this data, it calculates the potential biogas production and provides an estimate of the thermal energy that could be generated through anaerobic digestion.

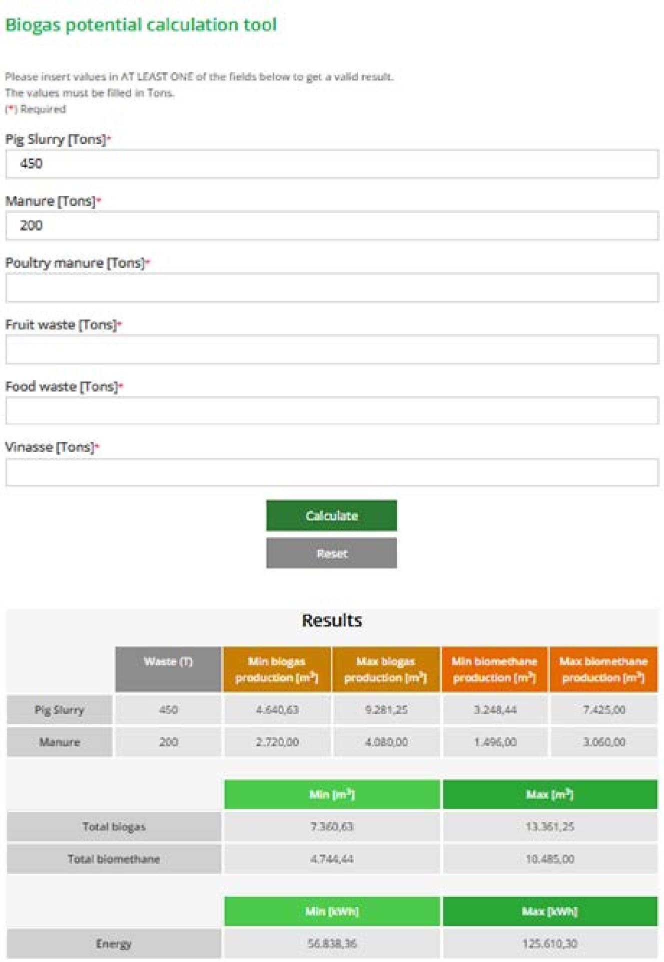

Figure 20. Screenshot of Twin 4.0

The Twin 4.0 tool offers several advantages and disadvantages. On the positive side, it is freely accessible and provides a practical, simple approach for users. However, it also has some limitations: the results it provides are estimative, it only considers certain predefined waste types, and it does not take economic factors into account.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

UniSQ Calculator for Australian Biogas

Developed by the University of Southern Queensland (UniSQ), a public Australian institution recognised for its focus on applied research and the development of sustainable solutions for the agro-industrial sector, this online tool (https://biogas.unisq.edu.au/calculator/#!/calculate) allows users to estimate the energy potential of various organic wastes.

The calculator estimates the potential volume of methane, energy content, and possible electricity generation based on the amount of waste entered. It also provides key economic indicators to support decision-making in bioenergy projects.

Users can select from various raw materials and enter the annual volume in tons of fresh matter (Figure 21). For greater accuracy, the percentage of total solids can also be adjusted based on known or measured values. As a result, the tool delivers a detailed estimate of annual methane production.

Types of feedstock

Type	TS (% FM)	Amount (tonnes/y)	Methane (m^3/y)	Remove
Pig slurry	6	500	5,990	✖
Sunflower straw	87	210	42,650	✖

Methane per year: 6,397 m^3

Now enter current costs for electricity and heat. They will be used to estimate the cost savings based on biogas production. Note that maintenance, operation, financial costs and additional benefits (sale of organic fertilizer, carbon credits etc) are not included in the economic calculation.

Results

Electricity (AUD/kWh) *	0.12
Heating (AUD/kWh) *	0
Energy total (kWh per year)	484,943
Capacity of CHP (kW_{el})	**19**
Electricity generation (kWh per year)	141,596
Heat generation (kWh per year)	209,496
Value of electricity and heat produced per year (AUD)	16,991
Investment costs (AUD)	258,596
Payback (years)	**15.2**

Now enter current costs for electricity and heat. They will be used to estimate the cost savings based on biogas production. Note that maintenance, operation, financial costs and additional benefits (sale of organic fertilizer, carbon credits etc) are not included in the economic calculation.

The measurement of the methane potential of individual residues/feedstocks using biomethane potential (BMP) tests can provide more accurate information. The National Centre for Engineering in Agriculture, at the University of Southern Queensland provides consultancy for companies and institutions interested in biogas feasibility studies or to support the operation of an existing biogas plant. This includes feasibility studies concerning wastewater treatment, wet and dry fermentation, biogas utilisation and digestate handling including its use as biofertiliser.

The focus of the biogas research includes:

- Assessment of different waste streams using BMP tests via the AMPTSII system (Bioprocess Control, Sweden);
- Evaluation of their suitability for anaerobic treatment;
- Process monitoring and control, as well as improvement of processes via a suite of semi-continuous and continuous bioreactors

This allows for a greater understanding of:

- Determining the suitability of a potential feedstock for biogas production
- Defining the suitable organic loading rate and establishing retention time for a given feedstock; and
- Designing suitable feedstock schedules.

Figure 21. Screenshot of UniSQ Calculator for Australian Biogas

The UniSQ Biogas Calculator for Australian biogas has several advantages and disadvantages. On the positive side, it operates at a high technical level, being based on scientific research and experimental data. It considers a wide variety of waste types and not only calculates the volume of biogas, but also its equivalent in electricity, heat, and total energy. The tool features a clear and educational interface and can provide a potential economic estimate. However, it also has limitations: its results are approximate, it does not account for legal or logistical variables, and it is specifically based on Australian conditions.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

Growdiesel Calculator

Developed by Growdiesel Ventures Limited, an Indian company specialised in renewable energy and a pioneer in converting organic waste into sustainable biofuels. Founded in 2006, the company has developed proprietary technologies to transform various types of waste into clean energy sources.

The tool (https://calculator.growdiesel.com/) consists of a biogas calculator that allows users to estimate the potential biogas production from different types of organic raw materials. Users can input the daily amount of raw material (in tons per day), the percentage of convertible volatile solids, and the gas price per cubic meter (Figure 22).

As a result, the tool provides detailed estimates of the biogas composition, both in its raw and refined form, offering a comprehensive view of the energy potential of the entered waste materials.

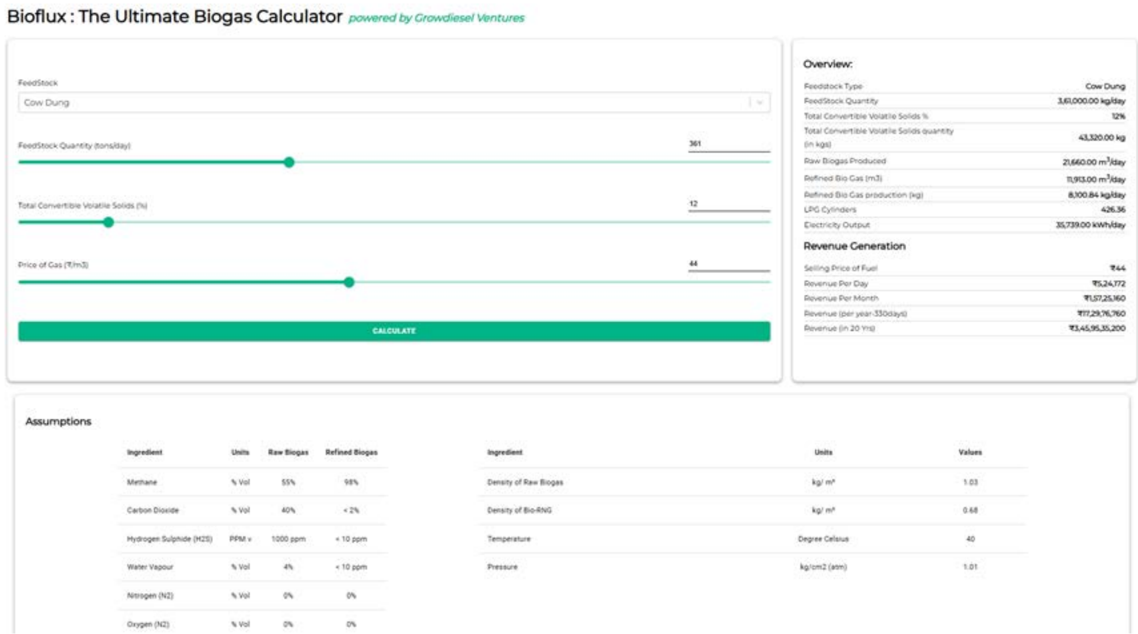

Figure 22. Screenshot of Growdiesel Calculator

The Growdiesel Calculator offers several advantages, including its ease of use, the ability to deliver fast and detailed results, and the fact that it is free and available online. It supports a wide variety of raw materials, making it versatile for different applications. However, it also has notable disadvantages. The results it provides are only estimations, and it does not allow for the calculation of mixed waste types. Additionally, it does not account for operating costs and displays prices exclusively in Indian Rupees, with no option to change the currency.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

Biteco Biogas

Developed by BITECO Biogas Ltd., a Ukrainian company founded in 2013, specialising in the turnkey construction of state-of-the-art industrial biogas plants. The company is dedicated to transforming organic waste into electricity, heat, biomethane, and organic fertilizers, offering comprehensive solutions that cover everything from technical design and construction to operation, maintenance, and modernisation of existing plants.

The tool (https://biteco-energy.com/es/biogas-calculator/) is a free and easy-to-use online calculator that allows users to first select a group of raw materials, then the specific type of waste, and enter the available quantity in tons per day (Figure 23).

Based on this data, the calculator estimates biogas production, electricity generation, and thermal energy output, providing a clear view of the energy potential of the selected waste materials.

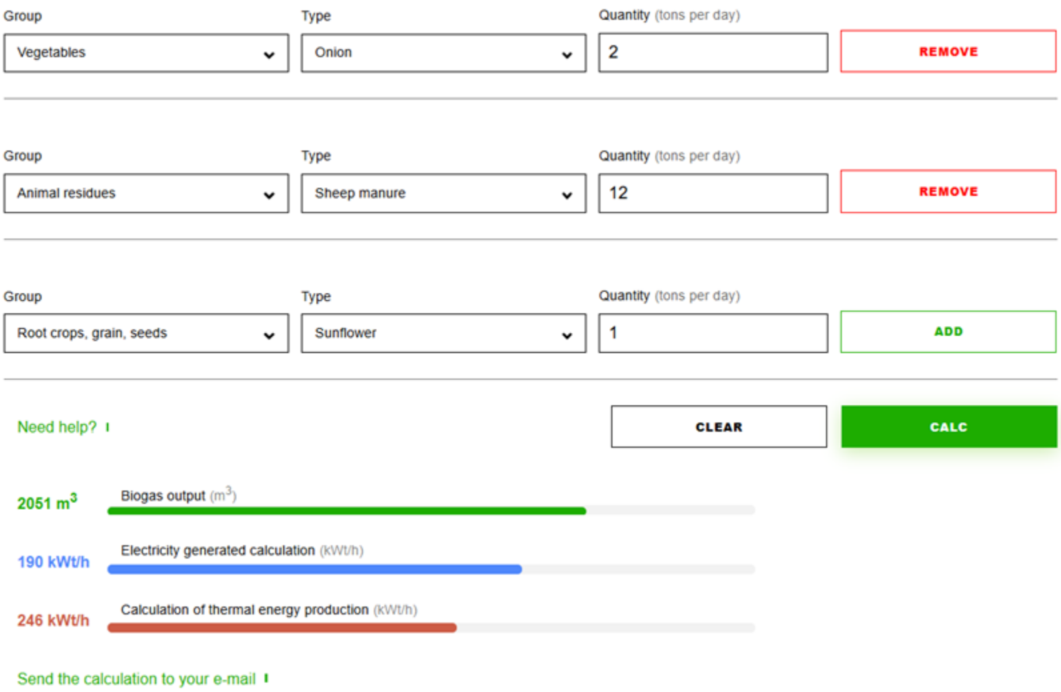

Figure 23. Screenshot of BITECO

The Biogas Calculator offers several advantages, including a simple and fast interface, the ability to handle a wide variety of raw materials, and the option to mix different waste types. However, it also presents some disadvantages, such as providing only general and approximate results and lacking detailed economic information.

Application and integration within the project

This tool can serve as a reference or benchmark to validate the results obtained during the future development of the AVPP. No support has been found for Python-based workflows or API access. The methodologies will be analysed and could be replicated in custom Python modules to ensure seamless interoperability within the AVPP framework.

PyADM1

PyADM1 is an open-source Python library [43] designed to simulate the dynamic behaviour of anaerobic digestion reactors (biogas plants), based on the internationally recognised Anaerobic Digestion Model No. 1 (ADM1)[44]. The tool (https://github.com/CaptainFerMag/PyADM1) offers a reproducible and modifiable implementation of ADM1 in Python, adapted from the Benchmark Simulation Model No. 2 (BSM2)[45], and has been validated through ring testing against the Matlab implementation of BSM2. PyADM1 is distributed under the MIT License and can be freely used, modified, and integrated within Python-based workflows.

To use PyADM1, Python 3 and supporting libraries (SciPy, NumPy, and copy) are required. The model uses input files to define the influent (feed) characteristics and reactor initial conditions (inoculum). Users can modify influent states, initial states, simulation time, time steps, and model parameters to reflect experimental data or specific scenarios.

PyADM1 is built in Python and relies on scipy.integrate.solve_ivp to solve the system of differential equations that define ADM1. The model implements both ODE and DAE formulations of ADM1 and allows users to switch between them using a DAE_switch parameter. The model equations describe disintegration, hydrolysis, acidogenesis, acetogenesis, and methanogenesis, along with ion and gas phase states. The code structure includes core functions such as ADM1_ODE, simulate, and DAESolve, which execute stepwise integration and calculation of state variables.

The simulation generates a results file (dynamic_out.csv) containing time series of the ADM1 state variables. These include soluble and particulate matter concentrations, ion species, pH, and gas phase components such as hydrogen (H_2), methane (CH_4), and carbon dioxide (CO_2). The file also records intermediate compounds (e.g., volatile fatty acids, acetate, propionate) and associated ionic forms, providing a comprehensive view of the biochemical and physicochemical dynamics within the anaerobic digester.

Application and integration within the project

In the context of the HarvRESt project, PyADM1 could be integrated into the AVPP framework to simulate anaerobic digestion and estimate biogas production based on defined feedstock and operational conditions. Its open-source Python implementation enables direct incorporation into automated workflows, offering flexibility for customised modelling and scenario testing relevant to virtual power plant simulations.

Biogas Plant Model (BiPMo)

BiPMo (Biogas Plant Model) is a Python-based simulation framework [46] developed for the optimal operation of biogas plants (https://github.com/tum-ens/bipmo). It works both as a standalone tool and in combination with the FLEDGE simulation environment. BiPMo is released under an MIT License, which allows free use, modification, and redistribution.

BiPMo is configured through CSV files that define different aspects of the biogas plant system. These include scenario settings (simulation period, temperatures, substrate availability, model type, and constant power/heat needs), feedstock characteristics (dry matter, volatile solids, biogas yield, methane content, and digestion time), CHP unit data (efficiencies, capacity, ramp rate, and power factor), gas storage parameters (state of charge limits), and price time series (price type, timestamp, and value). Together, these inputs establish the operational constraints, market conditions, and process parameters required for simulating plant performance.

BiPMo offers two model configurations: a simple model, which assumes constant biogas production without digestion dynamics, and a flexible model, which incorporates digestion process modelling and optimised feeding schedules. Using optimisation solvers like Gurobi or CPLEX, it generates operational schedules for biogas production, CHP operation, and storage management. The outputs include biogas production data (both instantaneous and cumulative), optimisation results such as CHP operation schedules, and automated plots visualising plant performance over time.

This structure and output make BiPMo particularly suitable as a reference framework or as a component for integration into the HarvREStt AVPP environment. Its Python-native design and CSV-driven configuration align well with the requirements of automated and flexible scenario modelling.

Application and integration within the project

Within the HarvREStt project, BiPMo could be used to simulate and optimise the operation of biogas plants under different scenarios, directly supporting the AVPP module. Its ability to model both simple and flexible plant configurations, combined with solver-based optimisation, allows assessment of operational strategies such as feeding schedules and CHP dispatch. The outputs—biogas production rates, cumulative generation, and CHP schedules—could be considered to be integrated into AVPP workflows to evaluate plant performance, system interactions, and flexibility contributions within a virtual power plant context.

3.5.2 *Hydrogen Production*

Holistic Hybrids Optimisation and Design Tool (H2Integrate)

H2Integrate (Holistic Hybrids Optimisation and Design Tool) is an open-source Python package [47] developed by NREL to model and design hybrid energy systems that produce electricity, hydrogen, ammonia, steel, and other products. It emphasises component-level modelling (e.g. electrolysers, renewables, storage), modularity, and extensibility in Python, built on NASA's OpenMDAO framework [48] for multidisciplinary modelling and optimisation (Figure 24). The project is publicly available under an open-source license, allowing free use and integration in Python-based energy system workflows (license details available in the GitHub repository [49]). It is currently under active development and may lack features formerly present; version 0.2.0 was the last to use the previous framework.

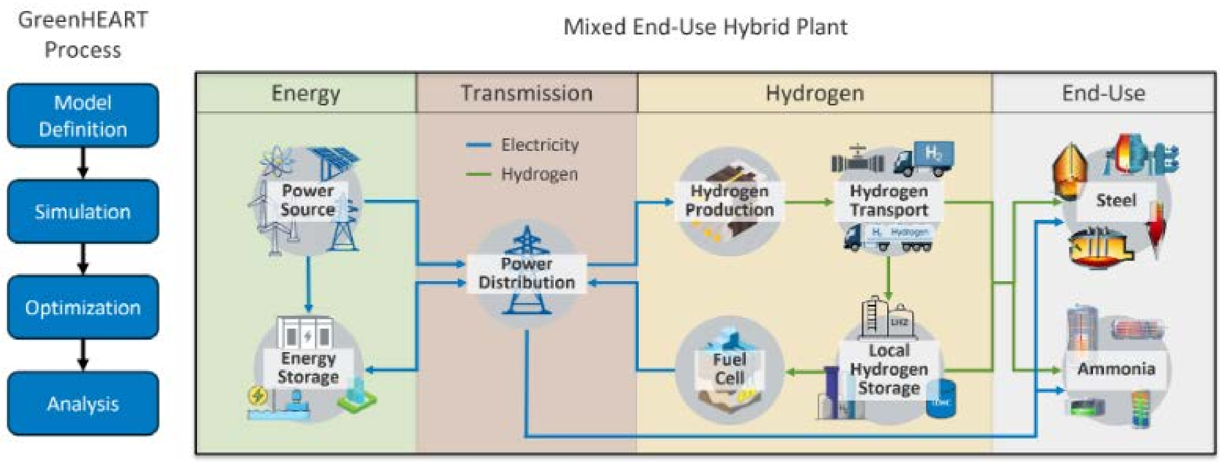

Figure 24. Example of energy system in H2Integrate [47]

H2Integrate is configured through user-readable YAML files that define the hybrid system architecture. The top-level configuration file specifies the analysis type and links to the driver, technology, and plant configuration files. These files collectively describe renewable generation profiles (such as wind and solar), system components including electrolysers, batteries, and other technologies, their performance and cost models, and how they are interconnected within the system. Analyses are typically conducted over a full year using hourly timesteps (8,760 hours), enabling detailed assessment of system design and operation.

When running H2Integrate, simulation results for the overall system and individual technologies are generated automatically. At the end of a simulation, the tool prints a list of all model inputs and outputs to the terminal, including variables such as electricity, hydrogen production, equipment efficiencies, and capital or operational costs. The raw numerical results are also stored within the Python prob object, which acts like a dictionary containing all model variables and their values. These results can be accessed programmatically using methods like get_val(), enabling users to retrieve outputs such as total hydrogen production in specified units or other detailed component-level data for further analysis and postprocessing.

Application and integration within the project

H2Integrate, as an open-source Python library, could be integrated within the project framework to provide hydrogen production estimation based on renewable generation profiles. Its modular design and built-in component models make it suitable for assessing electrolyser performance and hydrogen output, while also offering optimisation and system design capabilities if required in future stages of the AVPP development. This flexibility allows it to serve both as a dedicated tool for hydrogen estimation and, if needed, as a foundation for more advanced hybrid energy system optimisation within the AVPP context.

Optimised Hybrid Systems for Energy and Market management (oHySEM)

oHySEM (Optimised Hybrid Systems for Energy and Market management) is an open-source Python library[50] developed by the Instituto de Investigación Tecnológica at Universidad Pontificia Comillas. It is designed to optimise hybrid renewable electricity-hydrogen systems, particularly hybrid virtual power plants (H-VPPs), in short-term electricity markets (Figure 25). It supports both grid-connected and standalone systems, enabling integration of renewable generation, hydrogen production/storage, and battery energy storage into integrated scheduling and planning workflows . oHySEM is released under the GNU General Public License v3

(GPL-3.0), ensuring it is free software with copyleft terms. This licence allows full use, modification, and distribution, provided derivatives remain under the same license.

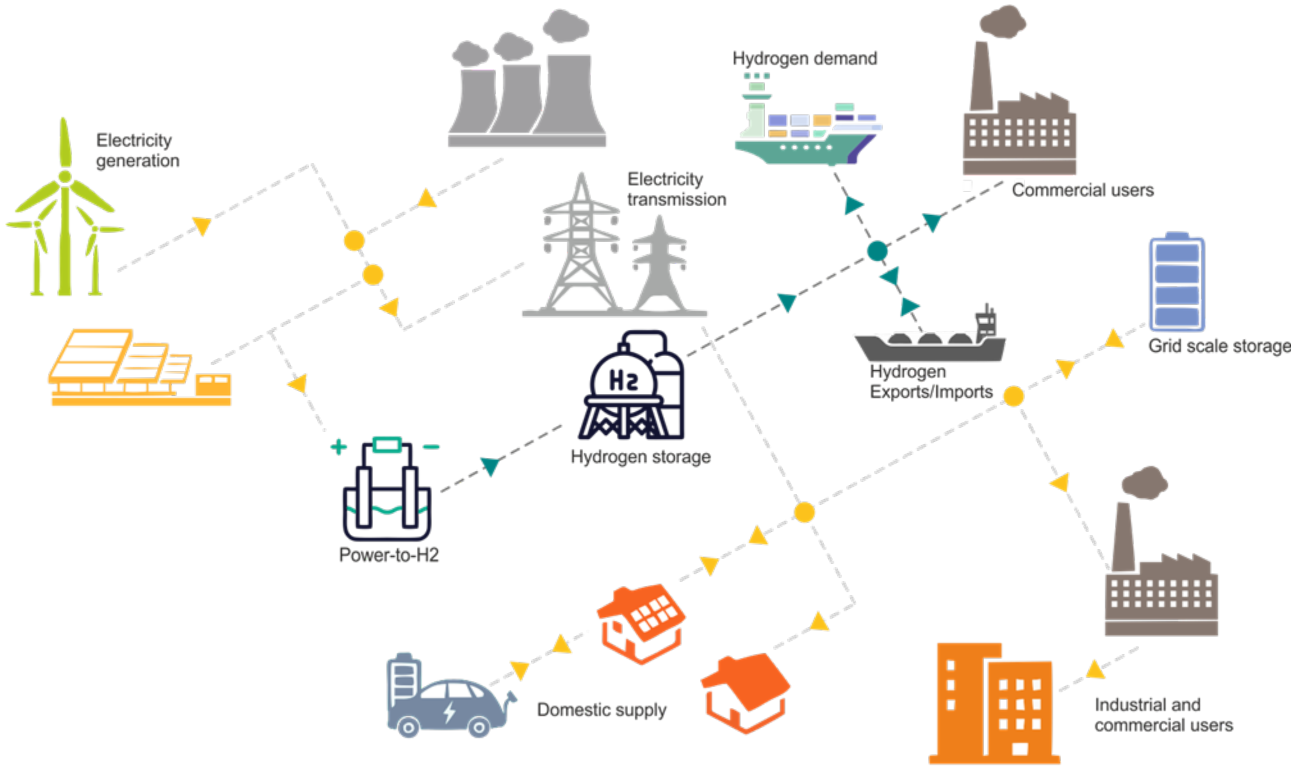

Figure 25. Integration of RES like Solar PV, Wind, and CCGT with hydrogen-based energy storage and battery systems in oHySEM [50]

oHySEM ingests case-specific CSV files—including time-series of renewable (solar PV, wind), conventional (e.g. CCGT) generation and demand; hydrogen system parameters such as electrolyser efficiency, capacity limits, and storage specifications; battery storage definitions and technical limits; plus market price series for electricity (day-ahead, intraday, real-time) and, if available, hydrogen markets. The library then constructs a Mixed-Integer Linear Programming model in Python using Pyomo to jointly optimise electricity flows, hydrogen production/storage via electrolysis, battery dispatch, and market bidding decisions across electricity and hydrogen markets. Temporal resolution is configurable between 15-minute and 1-hour intervals, and planning horizons range from one day up to 365 days—supporting both short-term operational scheduling and medium-/long-term decisions on asset sizing and location.

About the outputs, oHySEM generates a comprehensive suite of output CSV files and optional visual visualisations that include optimised dispatch schedules (daily/hourly) for renewables, batteries, and hydrogen electrolysis/storage; detailed market participation metrics such as day-ahead, intraday, real-time bidding, reserve offers, and imbalance settlement; recommendations for asset sizing and location planning of new hydrogen storage units, batteries, and generation capacity; and key financial indicators including energy sale revenues, imbalance penalties, and overall system value. All results are delivered in easy-to-process CSV format—including derived/transformed values, optimisation variable values, and dual values for constraints—and may be supplemented with graphical summaries for deeper insight.

Application and integration within the project

oHySEM naturally supports integration into an Advanced Virtual Power Plant architecture and, being an open-source Python library, it can seamlessly fit within the AVPP development framework. By defining existing assets (renewables, batteries, electrolysers) and optionally adding new ones, oHySEM enhances the platform with built-in optimisation algorithms that manage unified operational scheduling, market bidding strategies for energy and imbalances, and planning guidance for asset sizing and location. Its specific focus on hydrogen production, storage, and integration provides additional capabilities for advanced energy management. Additionally, it could be used as a tool for estimating hydrogen generation in simplified scenarios. This makes oHySEM a potential decision-support tool to be considered for future AVPP development, offering capabilities that could contribute to both day-to-day operational control and long-term asset planning while supporting efficiency and profitability through dynamic optimisation.

3.5.3 *Hybrid energy planning tools including storage*

In this section we discuss open-source frameworks, which are widely used in the energy modelling community. While our previous focus did not include storage within the optimisation loop, these tools explicitly allow storage technologies (such as batteries or pumped-hydro) to be modelled as investable elements, fully integrated into the system-wide cost-minimising optimisation problem.

Python for Power System Analysis (PyPSA)

PyPSA [51], [52] is an open source toolbox for simulating and optimising modern power and energy systems that include features such as conventional generators with unit commitment, variable wind and solar generation, storage units with efficiency losses and hydro inflow/spillage, coupling to other energy sectors, and mixed alternating and direct current networks (Figure 26). PyPSA is designed to scale well with large networks and long-time series. PyPSA is licensed under the open source MIT License.

component	list_name	description	category
Bus	buses	Fundamental node where all components attach.	
Carrier	carriers	Energy carriers of buses (e.g. "AC" for alternating current, "DC" for direct current, "hydrogen", or "heat") or technologies of other components (e.g. "wind", "gas turbine", "electrolyser", or "heat pump")	
Generator	generators	Power generator for the bus carrier it attaches to.	controllable_one_port
Load	loads	Loads represent a demand at the bus they are connected to (e.g. PQ power consumer).	controllable_one_port
Link	links	Links are used for controllable directed flows between two or more buses with arbitrary energy carriers (e.g. HVDC links, converters, conversions between carriers).	controllable_branch
Store	stores	Stores provide fundamental inter-temporal storage functionality not limited in charging or discharging power.	controllable_one_port
StorageUnit	storage_units	Storage units enable inter-temporal energy shifting with fixed nominal-energy-to-nominal-power ratio.	controllable_one_port

Figure 26. PyPSA. Sample of components [67]

Application and integration within the project

PyPSA could be considered for integration in the project because it is fully compatible with our development environment and supports modelling storage—including batteries and pumped-hydro—as investable technologies within a system-wide cost-minimising optimisation. It co-optimises dispatch and capacity planning for generation, conversion, storage, and transmission infrastructure, treating storage similarly to generation in the optimisation loop (starting from user-supplied renewable generation and demand series). Furthermore, PyPSA includes additional functionalities—such as sector coupling, detailed AC/DC network modelling, and the ability to incorporate non-convex constraints—that may align well with the development of a holistic hybridisation tool.

oemof.solph

Part of the open energy modelling framework, oemof.solph [53], [54] is a model generator for energy system modelling and optimisation using LP or MILP via Pyomo. It supports investment mode that treats storage as an optimizable component alongside generation technologies. The official storage_investment example demonstrates full system optimisation—including PV, wind, gas and storage capacities simultaneously—to achieve least-cost generation and storage mix (Figure 27). Oemof is licensed under the open source MIT License.

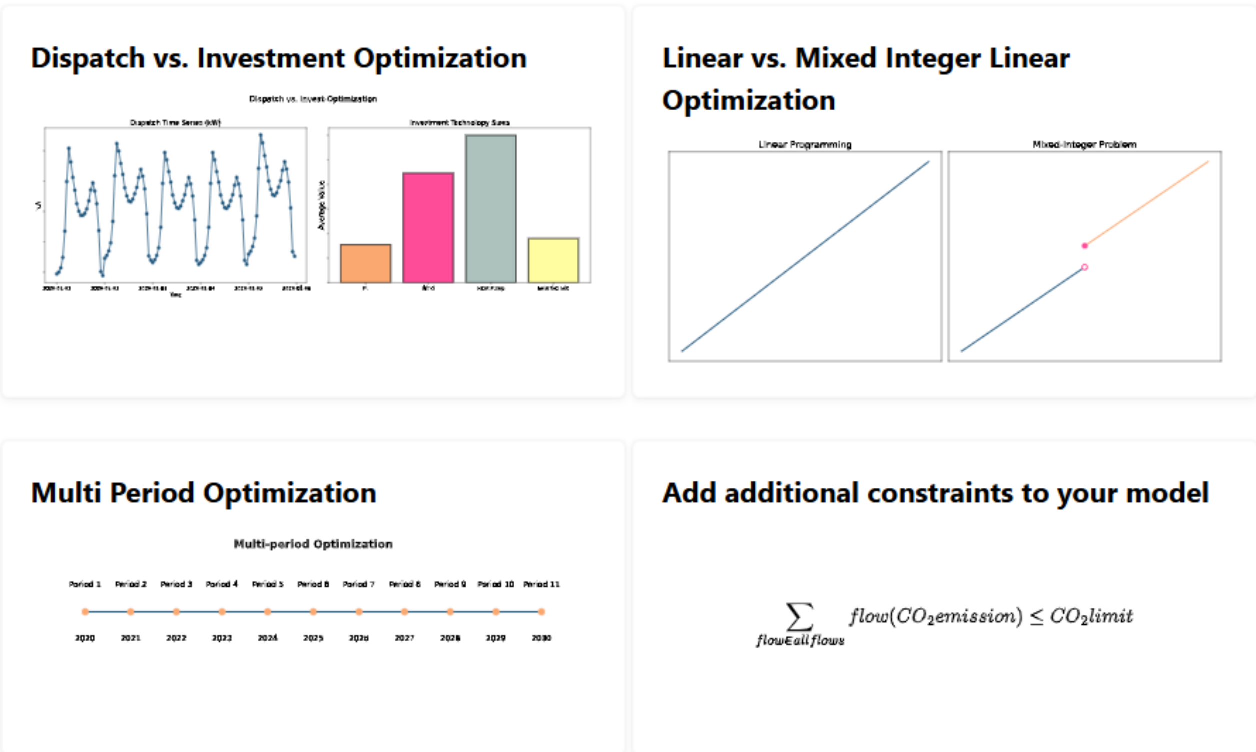

Figure 27. Optimisation capabilities in oemof.solph [68]

Application and integration within the project

oemof.solph could be considered for integration in the project because it is compatible with our development environment and supports modelling storage—including batteries and pumped-hydro—as investable technologies within the global cost-minimising optimisation problem. Its official storage_investment example demonstrates capacity optimisation for a system that includes storage alongside generation technologies, showing how storage can be treated as another component in the optimisation framework. Furthermore, oemof.solph includes functionalities such as sector coupling, network modelling, and the use of linear or mixed-integer constraints, which may align well with the development of a holistic hybridisation tool.

3.6 Identified Gaps and Technological Challenges

About the main gaps detected and technological challenges:

- The estimation of typical wind years in combination with solar data remains an unresolved issue. While there are applications that provide both wind and solar radiation data, wind speed is not prioritised to the same extent as solar irradiance. Moreover, coherence between wind and solar datasets is essential; typical wind and solar years obtained separately using different methods cannot be used together reliably. This highlights a gap that will be addressed within the project by equipping the developed tools with this capability.
- The most comprehensive hybridisation tools available are commercial, lacking the necessary flexibility and not natively supporting integration within a Python environment to build an ad-hoc solution. Consequently, the selection of tools to be used in the technological developments of the project is constrained by this significant limitation.
- In the case of biogas, most of the available tools are web-based, closed applications that cannot be integrated into the AVPP workflow (with some exceptions, such as PyADM1). Therefore, to meet the technical requirements of the AVPP, it is likely that the development of a custom module using conversion factors will be pursued.
- In the case of solar, wind, and hydro generation, the libraries pvlib, windpowerlib, and hydrogenerate, respectively, have been identified as promising and well-aligned with the requirements of the AVPP. These libraries (with some modifications) can serve as a solid foundation for further development.
- In the case of demand modelling, several alternatives will be evaluated depending on the feasibility of integrating the identified models into the AVPP working environment. The option of having preloaded typical hourly profiles might be considered a viable alternative solution.
- In case of agricultural modelling, the main gaps and challenges across the analysed tools relate to restrictive or copyleft licensing (APSIM, PCSE), lack of formal or mature Python APIs (APSIM, DSSAT official), incomplete SaaS/cloud readiness (APSIM, PASE), and integration hurdles due to packaging or rigid I/O structures (PASE). From a purely technical and licensing standpoint, the most suitable tools for the AVPP project appear to be AquaCrop-OSPy, PASE and PCS, with AquaCrop-OSPy excelling in ease of Python/web integration, PCSE offering flexibility, and PASE already combining crop and PV simulations in one framework, albeit with packaging refinements needed before seamless deployment.
- In the case of meteorological data collection, an efficient method for downloading historical data from any location worldwide is required. In this regard, Open-Meteo stands out for its quick response and ease of use. Regarding the generation of a representative year (TMY) to be used in hybridisation

models (and to reduce computational requirements), it will be necessary to develop a module capable of generating TMYs suitable for both wind and solar assets.

- Regarding the integration of all renewable generation sources and the simulation tools, along with potential demand profiles, storage options, energy flow definition, and scenario-based optimisation: Although there are very comprehensive commercial tools available (e.g., HOMER), they do not allow for straightforward integration into the project's working environment and do not cover all required features (e.g., agro modelling). For this reason, it initially appears necessary to develop a dedicated tool that encompasses all these basic functionalities while remaining open to future extensions and enhancements.

Some Risks or potential deviations have been detected:

- Limited Interoperability of commercial tools with ad hoc developments:
  - This will be addressed through integration layers and interface harmonisation in Task 6.1.
- Lack of real data to train/test models, libraries, and commercial software:
  - To mitigate this, we will use open-access datasets and ensure that the tools adopt a more generalist design, making them applicable beyond the HarvRESt use cases.

4. RES TECHNOLOGIES LIBRARY DESCRIPTION

The development of a structured reference library of modelling resources constitutes a foundational step towards the implementation of the AVPP. This library compiles existing open-source tools and models, complemented by the identification of gaps requiring ad-hoc developments, to serve as a centralised knowledge base. Its primary purpose is to complement the hybrid model optimisation activities of Task 5.3 and facilitate their subsequent integration into the AVPP environment in Task 6.1.

A conceptual repository structure is proposed to organise these modelling resources and functions in a modular and extensible manner. Rather than being a standalone installable library, this framework is intended to serve as an architectural blueprint, guiding how tools and models could be integrated within the AVPP.

This approach provides several benefits:

- Consistency and modularity: By defining standards for model organisation and interfacing, the library enables coherent access and reuse of modelling resources.
- Scalability: It allows the progressive inclusion of additional RES technologies, storage solutions, and control models as the AVPP evolves.
- Interoperability: The structure aligns with the AVPP's architecture, supporting seamless interaction between energy system models, control strategies, and agricultural production simulations.
- Efficiency in collaborative development: It offers a common framework for developers to contribute functions or extensions, reducing duplication of effort and facilitating future WP5 and WP6 tasks.

Following the review of state-of-the-art RES simulation tools and relevant components, this reference library encapsulates a curated set of external open-source libraries and internally developed models. It focuses on RES generation technologies suitable for farm-level deployment and provides the necessary groundwork for their combination into hybrid systems and subsequent integration into the AVPP.

This conceptual library is part of Deliverable D5.2 "RES technologies library", under Task T5.2 "Library of renewable alternatives for agro-communities" within WP5 "Modelling and optimisation of renewable alternatives for agro-communities". It builds upon prior work from Task T5.1 and is designed as an internal reference for use within the HarvRESt AVPP developers rather than as a publicly distributable package.

Connection with subsequent tasks

This conceptual library and its proposed structure will be further developed and materialised within Task 6.1, where it will serve as the foundation for the implementation of the AVPP. Its design is intrinsically linked to the hybrid model development carried out in Task 5.3 and is directly connected to the multi-criteria selection of scenarios in Task 6.2, as well as to the interpretation and added value provided for decision-making through the DSS system in Task 6.3. By establishing this architectural blueprint in Task 5.2, the consortium ensures a seamless transition from model compilation and structuring to their operational integration within the AVPP environment, reinforcing the continuity and coherence across WP5 and WP6.

4.1 Conceptual Framework and Structure

The RES Technologies Library is conceived as a modular and extensible set of simplified models representing renewable energy technologies and related systems, including agricultural processes, energy demand, storage, etc. Its primary objective is to provide a coherent and standardised modelling framework that could be integrated within the AVPP platform to enable the simulation and analysis of hybrid energy–agriculture systems.

The library should be based on a common internal API that defines how models are structured, executed, and integrated within the AVPP backend. This API is not intended as a public web service, but rather as an internal programming contract that ensures all models follow a uniform structure. It ensures uniformity in the way models are defined, executed, and integrated.

It should specify a minimal set of functions including: input validation, based on predefined schemas with explicit units, temporal resolution, and acceptable ranges; execution of the model and simulation over a defined time domain, returning time series results and summary indicators; aggregation functions for producing results over different temporal scales (e.g., monthly, seasonal, annual); save and reload model configurations and results in a standard format, so they can be stored, shared, and reused later.

From an architectural perspective, the library could be organised into specific technology or functional areas:

- Renewable generation models covering PV, wind, and hydropower. Biogas and hydrogen production will also be evaluated. These models are generally implemented as wrappers around the selected open-source modelling tools (e.g., pvlib, windpowerlib, HydroGenerate, PyADM1, H2Integrate).
- Storage models covering electrical battery systems, with emphasis on simplified charge–discharge behaviour and efficiency modelling.
- Demand models generating synthetic load profiles or process externally provided demand data.
- Agricultural models (e.g. *AquaCrop-OSPy*, *PCSE/WOFOST*, *PASE*) simulating crop growth, water balance, and yields, supporting scenarios that combine agricultural and energy production such as agrivoltaic systems.
- Auxiliary modules managing data acquisition and transformation from external meteorological and geospatial providers (e.g. Open-Meteo, ERA5, PVGIS, Global Wind Atlas), including routines for temporal resampling, gap filling, and the generation of TMY.

Each model will be implemented as a class that clearly defines its required inputs and expected outputs in compliance with the internal API. For example:

- PV models require irradiance components (GHI, DNI, DHI), ambient temperature, wind speed, and module/inverter specifications; outputs include AC and DC power time series, loss breakdowns, and aggregated performance ratios.
- Wind models require wind speed at reference height, air temperature, and pressure, with outputs including hub-height wind speed, power production, and capacity factors.
- Hydropower models require time series of flow rate, hydraulic head, and turbine/generator parameters, producing power generation and efficiency profiles.

- Biogas models take feedstock composition and operating conditions, returning methane yield, energy production, and by-products.
- Agricultural models require meteorological variables, soil properties, crop parameters, and management practices, producing evapotranspiration, soil water content, biomass accumulation, and yield.

This structure will ensure consistency, interoperability, and scalability across all components of the AVPP, while maintaining flexibility to integrate new technologies and models in future development phases.

Table 1 provides a summary of the tentative and highest-rated tools to be included in the AVPP platform, along with their inputs and outputs, during the development of the HarvRESt project and in future extensions. Depending on the final scope achieved by the AVPP developed in Task 6.1, some of these tools could be incorporated.

A simplified interaction between the RES Technologies Library and the wider AVPP stack is proposed as follows: the user interface enables scenario definition and data upload; the backend's internal API validates inputs and invokes the relevant models from the library; long-running simulations are delegated to background workers; external data services provide the necessary environmental and technical datasets; and the resulting outputs are processed for visualisation, analysis, and export.

Table 1. Conceptual Inputs and Outputs

Proposed Technologies and Tools	Proposed Inputs	Proposed Outputs
PV: *pvlib*	GHI/DNI/DHI, T_amb, wind_speed, system geometry, module/inverter specifications	P_ac[t], losses, P_dc[t], annual metrics
Wind: *windpowerlib*	Wind _speed, Temp, Pressure, hub height, power curve	P_ac[t], hub-height wind speed (v_hub[t]), air density, simple wake losses
Hydro: *HydroGenerate*	Flow rate (Q[t]), head, turbine/generator parameters	P[t], turbine efficiency (η_turb), annual energy, optional techno-economic metrics
Biogas: *PyADM1, BiPMo, ad-hoc tools*	Simple: substrate mix → CH_4, electrical/thermal power with efficiencies; Dynamic: ADM1 states and gas composition (CH_4, CO_2, H_2)	As above, depending on model type
H_2: *H2Integrate, ad-hoc tools*	Electrical power input (P_el[t]), efficiency curve (η(P)), ramp rate limits	Hydrogen mass (kg_H_2[t]), energy consumed, KPIs
Storage (battery): *PySam storage module*	Charging power (P_in[t]), charge/discharge efficiencies (η_c/η_d), maximum energy (E_max), maximum power (P_max)	State of charge (SOC[t]), discharge power (P_out[t]), losses
Demand: *demandlib, LPG*	Synthetic profiles (*demandlib, LPG*) or provided series; harmonised across sectors	Time series demand by sector
Agriculture: *AquaCrop-OSPy, PCSE, PASE*	Meteorology, soil, crop, and management data	Crop evapotranspiration (ETc), leaf area index (LAI), soil water content, yield; supports agrivoltaic coupling

Folder Structure

To facilitate maintainability and scalability, the library will be organised within the AVPP backend codebase according to a proposed source layout, as shown below:

The Figure 28 shows a simplified conceptual structure of the RES library package within the AVPP backend. The core module contains common interfaces and data validation routines. The data module manages data acquisition from external providers and generic loading functions. The models module is organised by technology domain (agriculture, demand, hydropower, PVs, storage, wind) and includes specific model implementations and default parameter sets. The models coupling module contains cross-domain interaction logic. Additional folders (io, docs, frontend, tests) support input/output processing, documentation, user interface development, and testing.

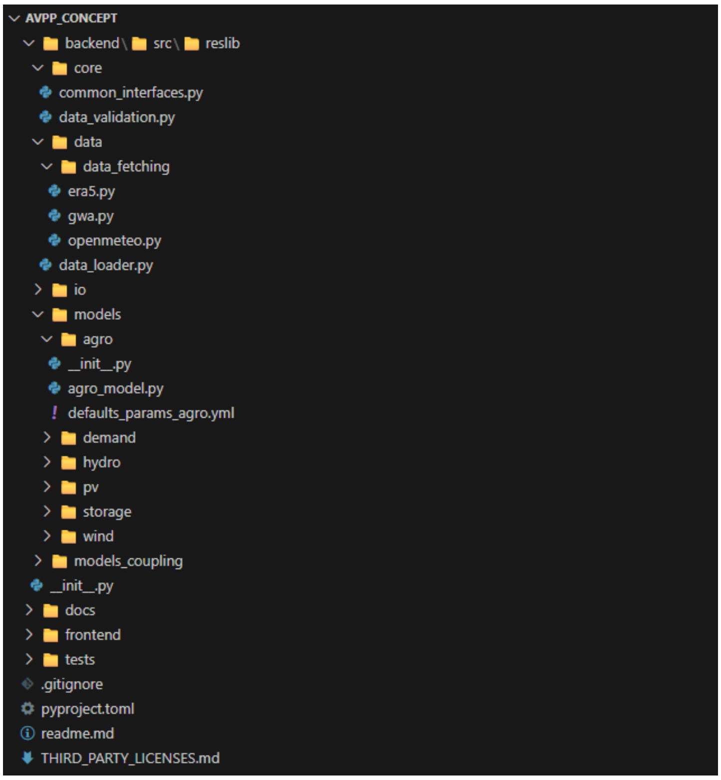

Figure 28. Conceptual RES technologies library folder structure within the AVPP

5. CONCLUSIONS

A comprehensive review and evaluation of available tools for renewable energy and agricultural modelling has been carried out to identify those capable of meeting the AVPP requirements defined in Task 6.1. This process has led to the pre-selection of several promising solutions, while also revealing important gaps and technological challenges.

Key challenges include the need for coherent wind–solar datasets for TMY generation, the limited Python integration and openness of the most complete hybridisation tools, and the reliance on closed, web-based platforms for biogas modelling. While libraries such as pvlib, windpowerlib, and hydrogenerate are well aligned with AVPP needs, agricultural modelling tools often present licensing constraints, incomplete Python APIs, or packaging issues. Meteorological data processing and the creation of representative years suitable for multiple RES assets remain open tasks. Furthermore, existing commercial hybridisation tools (e.g., HOMER) lack straightforward integration into the AVPP environment and do not address agro-modelling requirements, making the development of a dedicated, open, and extensible tool necessary.

Risks such as limited interoperability between commercial and custom developments, and the lack of real-world datasets for model validation, will be mitigated through interface harmonisation and the use of open-access data, ensuring general applicability beyond HarvREST use cases.

The tools shortlisted in this deliverable will form the technological foundation for the AVPP, both within the HarvRESt framework and in potential future expansions to incorporate additional RES types. In the next phase, Task 6.1 will initiate the AVPP development, addressing as many RES technologies as feasible while building an extensible and replicable structure. This structure will also be fully compatible with the MCDA module developed in Task 6.2, enabling multi-criteria decision-making to identify the most favourable scenarios according to stakeholder preferences. Furthermore, the AVPP will serve as a key input for the DSS in T6.3, allowing it to generate tailored recommendations based on the selected scenarios and the associated stakeholder priorities.

6. REFERENCES

[1] W. F. Holmgren, C. W. Hansen, and M. A. Mikofski, 'pvlib python: a python package for modeling solar energy systems', J. Open Source Softw., vol. 3, no. 29, p. 884, Sept. 2018, https://doi.org/10.21105/joss.00884

[2] K. S. Anderson, C. W. Hansen, W. F. Holmgren, A. R. Jensen, M. A. Mikofski, and A. Driesse, 'pvlib python: 2023 project update', J. Open Source Softw., vol. 8, no. 92, p. 5994, Dec. 2023, https://doi.org/10.21105/joss.05994

[3] S. Haas et al., wind-python/windpowerlib: Update release v0.2.2. (Feb. 20, 2024). https://doi.org/10.5281/zenodo.10685057

[4] B. Mitra, J. F. Gallego-Calderon, S. N. Elliott, T. M. Mosier, and C. J. Bastidas Pacheco, Hydrogenerate: Open Source Python Tool To Estimate Hydropower Generation Time-series, Version 3.6 or newer. (Oct. 2021). https://doi.org/10.11578/dc.20211112.1

[5] 'HydroGenerate Inputs / Outputs - General Workflow'. [Online]. Available: https://idaholabresearch.github.io/HydroGenerate/UserGuide_7_HydroGenerateWorkflow.html

[6] System Advisor Model (SAM). (2023). National Renewable Energy Laboratory, Golden, CO, USA. [Online]. Available: https://sam.nrel.gov

[7] PySAM v7.0.0. (July 17, 2025). National Renewable Energy Laboratory, Golden, CO, USA. [Online]. Available: https://github.com/NREL/pysam

[8] UL Solutions, 'HOMER - Hybrid Renewable and Distributed Generation System Design Software - Homer Pro'. [Online]. Available: https://www.homerenergy.com/products/pro/index.html

[9] UL Solutions, 'HOMER - Hybrid Renewable and Distributed Generation System Design Software - Homer Front'. [Online]. Available: https://www.homerenergy.com/products/front/index.html

[10] P. Lilienthal, 'Have it Your Way: The HOMER SaaS API', HOMER Microgrid News. [Online]. Available: https://microgridnews.com/way-homer-saas-api/

[11] NREL, 'HOMER: The Micropower Optimization Model'. 2004.

[12] D. P. Holzworth et al., 'APSIM – Evolution towards a new generation of agricultural systems simulation', Environ. Model. Softw., vol. 62, pp. 327–350, Dec. 2014, https://doi.org/10.1016/j.envsoft.2014.07.009

[13] D. Holzworth et al., 'APSIM Next Generation: Overcoming challenges in modernising a farming systems model', Environ. Model. Softw., vol. 103, pp. 43–51, May 2018, https://doi.org/10.1016/j.envsoft.2018.02.002

[14] APSIMInitiative/ApsimX. APSIM Initiative. [Online]. Available: https://github.com/APSIMInitiative/ApsimX

[15] J. Zhu et al., 'Developing perennial fruit crop models in APSIM Next Generation using grapevine as an example', Silico Plants, vol. 3, no. 2, p. diab021, July 2021, https://doi.org/10.1093/insilicoplants/diab021

[16] J. W. Jones et al., 'The DSSAT cropping system model', Eur. J. Agron., vol. 18, no. 3–4, pp. 235–265, Jan. 2003, https://doi.org/10.1016/S1161-0301(02)00107-7

[17] G. Hoogenboom et al., 'The DSSAT crop modeling ecosystem', in Advances in crop modelling for a sustainable agriculture, K. Boote, Ed., in Burleigh Dodds series in agricultural science, no. 75. , Cambridge Philadelphia: Burleigh Dodds Science Publishing, 2019. [Online]. Available: http://dx.doi.org/10.19103/AS.2019.0061.10

[18] G. Hoogenboom et al., Decision Support System for Agrotechnology Transfer (DSSAT) Version 4.8.5. (2024). DSSAT Foundation, Gainesville, Florida, USA. [Online]. Available: www.dssat.net

[19] DSSAT/dssat-csm-os. (Aug. 06, 2025). Fortran. DSSAT Foundation. [Online]. Available: https://github.com/DSSAT/dssat-csm-os

[20] X. He, XiaogangHe/pyDSSAT. (June 11, 2025). Python. [Online]. Available: https://github.com/XiaogangHe/pyDSSAT

[21] D. Quintero, daquinterop/Py_DSSATTools v3.0.0. (July 16, 2025). Python. [Online]. Available: https://github.com/daquinterop/Py_DSSATTools

[22] M. Salman, M. García-Vila, E. Fereres, D. Raes, and P. Steduto, The AquaCrop model: enhancing crop water productivity: Ten years of development, dissemination and implementation 2009-2019, 1st edn. in FAO Water Reports, no. 47. Rome: Food and Agriculture Organization of the United Nations, 2021. [Online]. Available: https://doi.org/10.4060/cb7392en

[23] S. De Roos, G. J. M. De Lannoy, and D. Raes, 'Performance analysis of regional AquaCrop (v6.1) biomass and surface soil moisture simulations using satellite and in situ observations', Geosci. Model Dev., vol. 14, no. 12, pp. 7309–7328, Nov. 2021, https://doi.org/10.5194/gmd-14-7309-2021

[24] L. Busschaert, S. De Roos, W. Thiery, D. Raes, and G. J. M. De Lannoy, 'Net irrigation requirement under different climate scenarios using AquaCrop over Europe', Hydrol. Earth Syst. Sci., vol. 26, no. 14, pp. 3731–3752, July 2022, https://doi.org/10.5194/hess-26-3731-2022

[25] Remote Sensing and Data Assimilation Research Group - KUL, KUL-RSDA/AquaCrop. (July 13, 2025). Fortran. [Online]. Available: https://github.com/KUL-RSDA/AquaCrop

[26] T. Foster et al., 'AquaCrop-OS: An open source version of FAO's crop water productivity model', Agric. Water Manag., vol. 181, pp. 18–22, Feb. 2017, https://doi.org/10.1016/j.agwat.2016.11.015

[27] T. D. Kelly and T. Foster, 'AquaCrop-OSPy: Bridging the gap between research and practice in crop-water modeling', Agric. Water Manag., vol. 254, p. 106976, Aug. 2021, https://doi.org/10.1016/j.agwat.2021.106976

[28] D. Raes, P. Steduto, T. C. Hsiao, and E. Fereres, 'Reference Manual, Chapter 1 – AquaCrop, Version 7.1', Rome, Italy: FAO, 2023. [Online]. Available: https://openknowledge.fao.org/handle/20.500.14283/br246e

[29] A. De Wit et al., ajwdewit/pcse: PCSE 5.4.0. (Sept. 25, 2018). Zenodo. https://doi.org/10.5281/ZENODO.1435175

[30] Allard de Wit, 'PCSE: The Python Crop Simulation Environment — Python Crop Simulation Environment 6.0 documentation'. [Online]. Available: https://pcse.readthedocs.io/en/stable/index.html

[31] European Commission, 'EUPL text (EUPL-1.2) | Interoperable Europe Portal'. [Online]. Available: https://interoperable-europe.ec.europa.eu/collection/eupl/eupl-text-eupl-12

[32] R. Bruhwyler et al., 'Modelling light-sharing in agrivoltaics: the open-source Python Agrivoltaic Simulation Environment (PASE 1.0)', Agrofor. Syst., vol. 98, no. 8, pp. 2747–2764, Dec. 2024, https://doi.org/10.1007/s10457-024-01090-8

[33]R. Bruhwyler, 'Open-Source Development of The Python Agrivoltaic Simulation Environment and Case Studies with PASE 1.0', PhD Thesis, ULiège - Université de Liège [Gembloux Agro-Bio Tech], Gembloux, Belgium, 2025. [Online]. Available: https://hdl.handle.net/2268/326176

[34]R. Bruhwyler, DEAL public / PASE · GitLab v1.2.0. [Online]. Available: https://gitlab.uliege.be/deal-public/pase/-/tree/v1.2.0?ref_type=tags

[35]'Contents — demandlib 0.2.3a1 documentation'. Accessed: Aug. 04, 2025. [Online]. Available: https://demandlib.readthedocs.io/en/latest/

[36]N. Pflugradt, P. Stenzel, L. Kotzur, and D. Stolten, 'LoadProfileGenerator: An Agent-Based Behavior Simulation for Generating Residential Load Profiles', J. Open Source Softw., vol. 7, no. 71, p. 3574, 2022, https://doi.org/10.21105/joss.03574

[37]'LoadProfileGenerator'. Accessed: Aug. 04, 2025. [Online]. Available: https://www.loadprofilegenerator.de//

[38]'Homepage | Copernicus'. Accessed: Aug. 05, 2025. [Online]. Available: https://www.copernicus.eu/en

[39]H. Hersbach et al., 'ERA5 hourly data on single levels from 1940 to present'. Copernicus Climate Change Service (C3S) Climate Data Store (CDS), 2023. https://doi.org/10.24381/cds.adbb2d47

[40]P. Zippenfenig, Open-Meteo.com Weather API. (2023). https://doi.org/10.5281/zenodo.7970649

[41]N. N. Davis et al., 'The Global Wind Atlas: A high-resolution dataset of climatologies and associated web-based application', Bull. Am. Meteorol. Soc., vol. 104, no. 8, pp. E1507–E1525, Aug. 2023, https://doi.org/10.1175/BAMS-D-21-0075.1

[42]T. N et al., 'Photovoltaics Geographical Information System: Status Report 2024', no. KJ-01-25-094-EN-N (online), 2025, https://doi.org/10.2760/2979595

[43]Peyman, CaptainFerMag/PyADM1. (July 30, 2025). Python. Accessed: Aug. 05, 2025. [Online]. Available: https://github.com/CaptainFerMag/PyADM1

[44]P. Sadrimajd, P. Mannion, E. Howley, and P. N. L. Lens, 'PyADM1: a Python implementation of Anaerobic Digestion Model No. 1', bioRxiv, 2021, https://doi.org/10.1101/2021.03.03.433746

[45]U. Jeppsson and C. Rosen, 'Aspects on ADM1 Implementation within the BSM2 Framework', Jan. 2006.

[46]'BiPMo – Biogas Plant Model — bipmo documentation'. Accessed: Aug. 05, 2025. [Online]. Available: https://tum-ens.github.io/bipmo/develop/index.html

[47]'H2Integrate - Holistic Hybrids Optimization and Design Tool — H2Integrate'. Accessed: Aug. 06, 2025. [Online]. Available: https://nrel.github.io/H2Integrate/intro.html

[48]J. S. Gray, J. T. Hwang, J. R. R. A. Martins, K. T. Moore, and B. A. Naylor, 'OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization', Struct. Multidiscip. Optim., vol. 59, no. 4, pp. 1075–1104, Apr. 2019, https://doi.org/10.1007/s00158-019-02211-z

[49]GitHub NREL/H2Integrate. (Aug. 05, 2025). PowerBuilder. National Renewable Energy Laboratory. Accessed: Aug. 06, 2025. [Online]. Available: https://github.com/NREL/H2Integrate

[50]'ohysem. Overview — version 1.0.14'. Accessed: Aug. 06, 2025. [Online]. Available: https://ohysem.readthedocs.io/en/latest/Introduction.html

[51]T. Brown, J. Hörsch, and D. Schlachtberger, 'PyPSA: Python for Power System Analysis', J. Open Res. Softw., vol. 6, no. 4, 2018, https://doi.org/10.5334/jors.188

[52]'Introduction — PyPSA: Python for Power System Analysis'. Accessed: Aug. 06, 2025. [Online]. Available: https://pypsa.readthedocs.io/en/latest/getting-started/introduction.html

[53]U. Krien, P. Schönfeldt, J. Launer, S. Hilpert, C. Kaldemeyer, and G. Pleßmann, 'oemof.solph—A model generator for linear and mixed-integer linear optimisation of energy systems', Softw. Impacts, vol. 6, p. 100028, 2020, https://doi.org/10.1016/j.simpa.2020.100028.

[54]'Oemof:Introductory Tutorials - v0.6.0'. Accessed: Aug. 06, 2025. [Online]. Available: https://oemof-solph.readthedocs.io/en/stable/introductory_tutorials.html

[55] Source: https://pvlib-python.readthedocs.io/en/stable/gallery/index.html (Accessed 13/08/2025)

[56] Source: https://windpowerlib.readthedocs.io/en/v0.2.2/getting_started.html#introduction (Accessed 13/08/2025)

[57] Source: https://idaholabresearch.github.io/HydroGenerate/DeveloperGuide_2_MethodsFunctions.htm l (Accessed 13/08/2025)

[58] Source: https://sam.nrel.gov/images/webinar_files/sam-webinars-2020-intro-to-sam.pdf (Accessed 13/08/2025)

[59] Source: https://www.homerenergy.com/products/pro/index.html (Accessed 13/08/2025)

[60] Source: https://dssat.net/models-overview/components/ (Accessed 13/08/2025)

[61] Source: https://pcse.readthedocs.io/en/stable/user_guide.html (Accessed 13/08/2025)

[62] Sources: https://oemof.org/2025/04/11/demandlib-0-2-2-vdi-and-bdew25/#more-3891 (Accessed 13/08/2025)

[63] Source: https://www.loadprofilegenerator.de/ (Accessed 13/08/2025)

[64] Source: https://open-meteo.com/en/docs/historical-weather-api (Accessed 13/08/2025)

[65] Source: https://globalwindatlas.info/en/ (Accessed 13/08/2025)

[66] Source: https://re.jrc.ec.europa.eu/pvg_tools/es/ (Accessed 13/08/2025)

[67] Source: https://pypsa.readthedocs.io/en/latest/user-guide/components.html (Accessed 13/08/2025)

[68] Source: https://oemof-solph.readthedocs.io/en/stable/optimization.html (Accessed 13/08/2025)

HarvRESt
Greener Farming with RES

The project

The HarvRESt project aims to enhance the sustainable production of renewable energy at farm-level. This approach not only makes farms climate-neutral but also optimizes production, reduces their impact on natural resources and biodiversity, and provides energy services to communities, thereby diversifying economic income. However, deciding how best to integrate renewable energy sources (RES) on a farm is not without its challenges. The decision is a complex one, with many factors to consider. Due to this, HarvRESt seeks to identify, understand, and overcome the existing barriers hindering the widespread adoption of this innovative approach. Current initiatives often overlook the complex interactions and factors within the farming and RES context, resulting in ineffective support for decision-making based on accurate projections, estimations, and forecasts. HarvRESt will therefore consolidate and enhance existing knowledge, creating an Agricultural Virtual Power Plant capable of running diverse scenarios and farm configurations. This tool will determine the best operational procedures for a given RES solution, providing valuable data to a decision support system. This system will weigh trade-offs and key indicators, offering tailor-made recommendations to farmers and policymakers.

PARTNER		SHORT NAME
circe Creating together	CIRCE Research Centre	CIRCE
Beta UVIC	BETA Technological Centre	UVic-UCC
NORCE	NORCE	NORCE
TECNOALIMENTI	Tecnoalimenti	TCA
WHITE	WHITE	WR
Suite5 We Deliver Intelligence	Suite5 Data Intelligence Solutions Ltd.	Suite5
EnGreen	EnGreen	EnG
ConTerra	ConTerra	CT

Confagricoltura	Confagricoltura	CONFAGRI
Solidale del Circeo	Fattoria Solidale del Circeo	FSDC
Viñas del Vero Somontano	Viñas del Vero	VdV
Viñedos del Rio Tajo	Viñedos del Rio Tajo	VRT
sorigué	Sorigué	ACSA
GGE	Grønn Gårdsenergi AS	GGE
Food & Bio Cluster Denmark	Food & Bio Cluster Denmark	FBCD
Climate KIC	Climate KIC	CKIC
HEC | S&O Climate & Earth center	HEC Paris	HEC

Contact us

www.harvrest.eu

https://linkedin.com/harvRESt

https://twitter.com/HarvREST_eu