LOCOMOTION

Decision-making, gamification and awareness tools

WP 11, Task 11.1, D.11.1

August 2023

LOW-CARBON SOCIETY: AN ENHANCED MODELLING TOOL FOR THE TRANSITION TO SUSTAINABILITY (LOCOMOTION)

H2020-LC-CLA-2018-2

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821105.

DOCUMENT HISTORY

Project Acronym	LOCOMOTION	
Project title	Low-carbon society: an enhanced modelling tool for the transition to sustainability	
Project coordination	Universidad de Valladolid (Spain)	
Project duration	1st June 2019 – 30th November 2023	
Deliverable No.	D11.1. Decision-making, gamification and awareness tools	
Dissemination level	Public (PU)	
Status		In progress
		To be verified by other WPs
	∗	Final
Due date of deliverable	31st of August 2023	
Delivery date	31st of august 2023	
Version	v.2.3	
Work package	WP11 – User engagement: Decision-making, awareness tools and gamification	
Lead beneficiary	CREAF	
Author (s)	Iván Ramos (CARTIF), Francisco Javier Miguel (CARTIF), Rafael Delgado (CARTIF), Manuel Pérez (CARTIF), Roger Samsó (CREAF), David Escudero (Uva), Yania Crespo (UVa)	

Date	Ver.	Contributors	Comment
29/11/2022	0.1	Iván Ramos Diez (CARTIF)	Table of contents
14/03/2023	1.1	Roger Samsó (CREAF)	Model Analyzer
16/05/2023	1.1	Iván Ramos Diez (CARTIF), Francisco Javier Miguel (CARTIF)	Methodology and user needs section. Model Explorer Section.
21/06/2023	2.1	David Escudero Mancebo (UVA)	Crossroads II section
21/06/2023	2.1	Yania Crespo (UVA)	Crossroads II section
24/07/2023	2.2	Roger Samsó (CREAF)	Synergies, future work and other edits
08/08/2023	2.2	Iván Ramos Diez (CARTIF)	Final version of user needs section. Model explorer and document review.
28/08/2023	2.2	Iván Ramos Diez (CARTIF), Rafael Delgado (CARTIF)	Model Explorer Section and final version of the document.

COPYRIGHT

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS AND ACRONYMS

Acronym	Description
WP	Work Package
BAU	Business As Usual
SSP	Shared Socioeconomic Pathways
CC	Climate Change
app	application

EXECUTIVE SUMMARY

In this deliverable we present all the work carried out during the development of Task 11.1 entitled *Decision-making, gamification and awareness creation* Throughout this document we present the work carried out to design, develop and deploy the three LOCOMOTION tools (Model Analyser, Global Sustainability Crossroads II Game and Model Explorer) based on WILIAM model and associated results.

The fists part of the document is dedicated to describe the methodology followed to implement the tools and also the consultation processes that are the basis to create the tool concept and also to start the design definition by means of mock-ups development.

In the document, each tool is described following the same structure to ensure a good comprehension of the different implementation steps. The initial part of each tool description, is related with the design, including all the views developed to translate the use cases into functional activities that the user could implement by using the tool. After that, the frontend and backend developments are described considering the architecture of each tool and the way in which WILIAM model an associated results are integrated. Each tool description concludes with a section describing the activities carried out to ensure the quality of the developed code and the deployment guide that will be useful to share the tool in the web allowing the dissemination as key project results.

Finally, the deliverable includes a section to explain all the pending issues that need to be covered before the final release of the three tools until the end of the project and also, some annexes as manuals to guide users in the first steps with the tool. These manuals could be updated with the pending implementations and could be improve by integrating videos on the use of the tool.

INTRODUCTION

This document presents the results of the activities carried out for the implementation of the LOCOMOTION tools. Together with the model and its translation into Python code, the results contained in this document, which shape the three project tools, may be required as the fundamental results of the project, since they will allow the dissemination of the model and policies for a sustainable transition to a greater number of stakeholder groups, which will be of great help to put in value the results of the project.

The implementation of the LOCOMOTION tools (Model analyzer, Model Explorer and the Global Sustainability Crossroads II Game) allowed to achieve three fundamental objectives of WP11, which are presented below:

1. Provide technical tools tailored to the different stakeholder groups to access to the models and their information.
2. Develop gamification tools for education purposes.
3. Provide a simplified version of the models for civil society.

Before starting with the developments carried out for the implementation of each tool, it is convenient to at least develop in a simple way what each tool is. A brief description of each tool is presented below.

The **Model Analyzer** is a desktop application, which allow users to configure scenarios and running them using the WILIAM model and the same parametrization as using the Vensim software. The expected target audience for this application are modellers and experts on the energy transition, and policy makers including their advisors.

The **Global Sustainability Crossroads II Game** is a cooperative and social role-playing game played in groups using electronic devices thanks to a web-based implementation of the tool. Designed for the educational community, the main idea behind the Crossroads II Game is to raise awareness about the climate transition and streamline player interaction in decision-making on the implementation of climate change policies through gamification.

The **Model Explorer** is a user-friendly and easy-to-use web-based application to explore WILIAM IAM without needing to be a modelling expert. With a reduced functionality, it main goal is to increase the awareness in the civil society in terms of climate change, helping in the definition of more sustainable future scenarios.

1. METHODOLOGY FOR TOOLS DEVELOPMENT

In this section, a description of the main steps to design, develop and deploy the LOCOMOTION tools are included. As part of software design, a set of steps should be followed. These steps are part of the Software Development Life Cycle (SDLC) methodology, which includes the application of standard business practices for building software applications. It is typically divided into six to eight steps including at least the following aspects: Planning, Requirements, Design, Build, Document, Test, Deploy and Maintain. The general flow using SDLC methodology with six steps for LOCOMOTION is represented in Figure 1. These general steps are described in depth below.

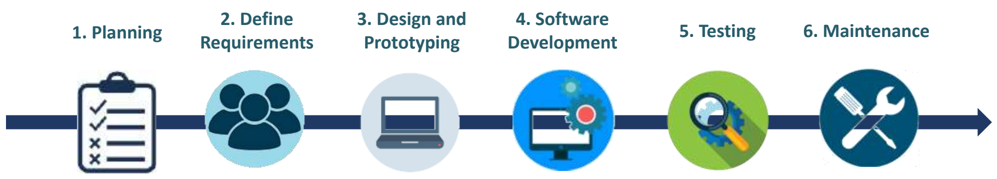


Figure 1. Main steps of SDLC methodology.

- **Planning**: This step defines the scope and purpose of the application considering the time and the efforts necessary to develop and deploy a functional tool.
- Define Requirements: It could be considered as part of the planning step. At the end, the requirements try to determine what the application is supposed to do according to user needs and the requirements considering functional and non-functional needs of the proposed concept idea of an application.
- **Design and Prototyping**: This step covers the way of a software application will work. Some aspects need to be considered: architecture, user interface, platforms, programming method, communications or security. In addition, prototyping could be useful to demonstrate the basic idea on how the application looks and works.
- **Software Development**: It is considered as the writing of the program (coding). The code of the program could be written by a single developer or by a team of developers. During the coding process are included many other tasks as finding and fixing errors and glitches or test and compile the code to run the application. The code can include some instructions and explanations to guide developer making easy development task. Finally, this step includes the development of user guides.
- **Testing**: The testing step helps to reduce the number of bugs and glitches that users encounter. This leads to a higher user satisfaction and a better usage rate. It is critical to test an application before making it available to users by end-to-end test covering all the parts of the developed application. Finally, after the testing, the application will be deployed to be available to users.
- **Maintenance**: Once the development cycle is finished, the application is being used in real environments. Users discover bugs that were not found during testing. These errors need to be resolved to have launched a new release of the application.

These steps of SDLC methodology previously described, are grouped in three main phases to achieve the development of the LOCOMOTION tools. The phases are: Characterization, User needs identification and tool design, and finally, Tool development (Figure 2). In each phase, we include the following steps of SDLC methodology:

- Characterization phase: Includes the “Planning” step.
- User needs identification and tool design: It groups the steps of “Define requirements” and, “Design and prototyping”.
- Tool development: This phase is the most critical one and refers to “Software development”, “Testing” and “Maintenance”.

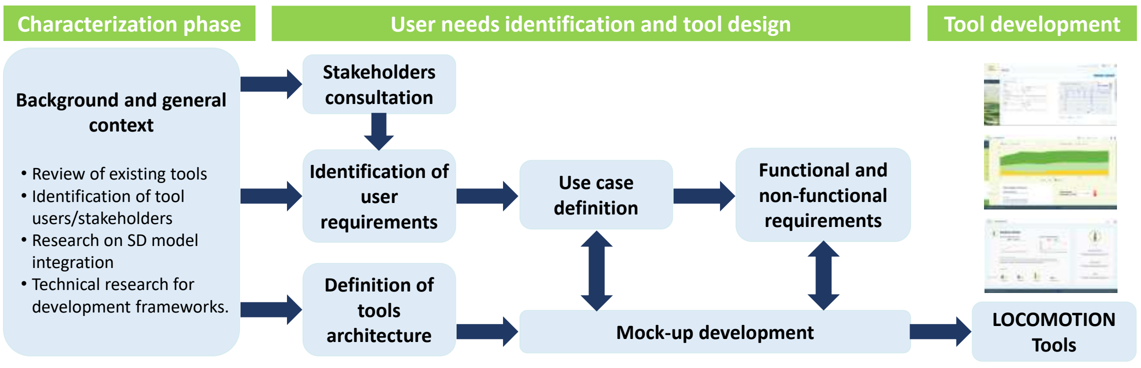

Characterization phase
User needs identification and tool design
Tool development
Background and general context
• Review of existing tools
• Identification of tool users/stakeholders
• Research on SD model integration
• Technical research for development frameworks.
Stakeholders consultation
Identification of user requirements
Definition of tools architecture
Use case definition
Functional and non-functional requirements
Mock-up development
LOCOMOTION Tools

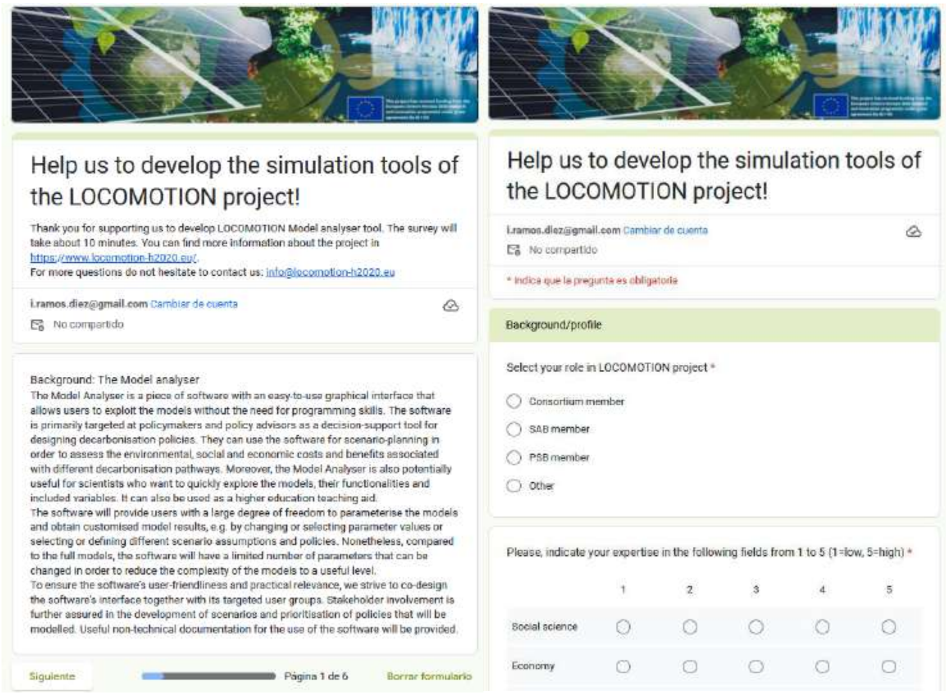
Figure 2. Main steps in the development of LOCOMOTION tools implementation.

2. IDENTIFICATION OF USER NEEDS

This section presents the results of the identification of different user needs that were used as the starting point for the implementation of LOCOMOTION tools. User needs are crucial due to they provide real insights to guide the design of the tools and also the selection of technologies and functionalities that will be covered by the tools. Having abundant information to guide the developments can be crucial to be able to obtain tools that continue to be used beyond the end of the project.

2.1. CONSULTATION PROCESS: SURVEY DEVELOPMENT

To obtain information on the user needs, three different online surveys were developed (one for each tool) in order to identify the main requirements for each stakeholder group and specific user. These surveys were distributed between the consortium partners and Scientific Advisory Board (SAB) members as experts in the modelling and policy implementation, and also with the stakeholder's panel as civil society and end-users related with climate change and sustainable development. Figure 3, presents an example of the surveys (Model analyser) carried out in the last part (October-November) of 2020.



Help us to develop the simulation tools of the LOCOMOTION project!

Thank you for supporting us to develop LOCOMOTION Model analyser tool. The survey will take about 10 minutes. You can find more information about the project in https://www.locomotion-h2020.eu/.
For more questions do not hesitate to contact us: info@locomotion-h2020.eu

i.ramos.diez@gmail.com Cambiar de cuenta
No compartido

Background: The Model analyser
The Model Analyser is a piece of software with an easy-to-use graphical interface that allows users to exploit the models without the need for programming skills. The software is primarily targeted at policymakers and policy advisors as a decision-support tool for designing decarbonisation policies. They can use the software for scenario-planning in order to assess the environmental, social and economic costs and benefits associated with different decarbonisation pathways. Moreover, the Model Analyser is also potentially useful for scientists who want to quickly explore the models, their functionalities and included variables. It can also be used as a higher education teaching aid.
The software will provide users with a large degree of freedom to parameterise the models and obtain customised model results, e.g. by changing or selecting parameter values or selecting or defining different scenario assumptions and policies. Nonetheless, compared to the full models, the software will have a limited number of parameters that can be changed in order to reduce the complexity of the models to a useful level.
To ensure the software's user-friendliness and practical relevance, we strive to co-design the software's interface together with its targeted user groups. Stakeholder involvement is further assured in the development of scenarios and prioritisation of policies that will be modelled. Useful non-technical documentation for the use of the software will be provided.

Help us to develop the simulation tools of the LOCOMOTION project!

i.ramos.diez@gmail.com Cambiar de cuenta
No compartido

* Indica que la pregunta es obligatoria

Background/profile

Select your role in LOCOMOTION project *

- ◯ Consortium member
- ◯ SAB member
- ◯ PSB member
- ◯ Other

Please, indicate your expertise in the following fields from 1 to 5 (1=low, 5=high) *

	1	2	3	4	5
Social science	◯	◯	◯	◯	◯
Economy	◯	◯	◯	◯	◯

Figure 3. Screenshots of Model analyser survey.

2.2. CONSULTATION PROCESS: SURVEY RESULTS

The total number of answers obtained for each questionnaire is presented below:

- Model analyser: 20 answers.
- Global Sustainability Crossroads II Game: 14 answers.
- Model Explorer: 23 answers.

In the following sections, the results obtained for each of the tools are analyzed in detail.

2.2.1. MODEL ANALYSER

In this section, the results of the questionnaire to collect the requirements from the stakeholders that will help as the basis to guide the development of the Model analyser is presented. Figure 4, presents the distribution of the sample among the different groups that are part of the project. The higher number of answers were collected from the consortium member followed by other stakeholders and SAB members.

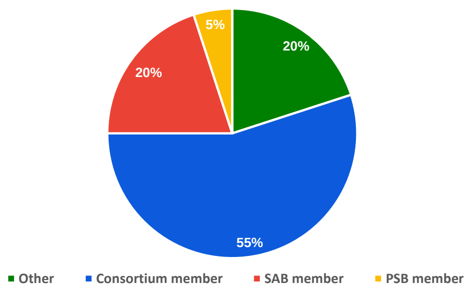


Figure 4: Role in LOCOMOTION project for Model analyser answers.

According to the field of expertise, a valuable number of the collected answers comes from experts on energy, environment and climate change adaptation and mitigation (Figure 5). They have experience as IAM developers and users being their work closely related with modelling tasks (Figure 6). The results of the model, are usually used to evaluate the challenge of the energy transition and also for the scientific research being the Microsoft software the most interesting for users to work with the app (Figure 7).

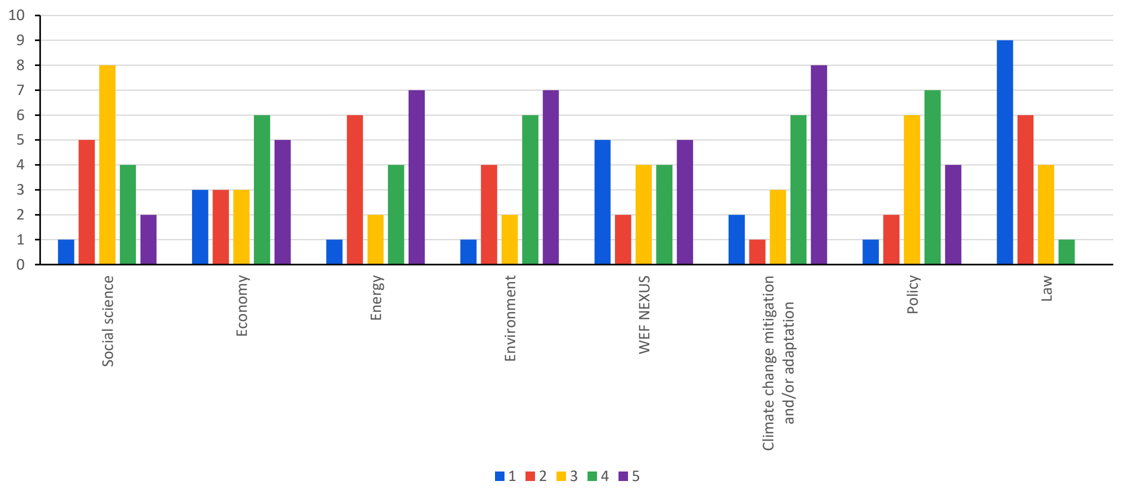


Figure 5: Expertise by field for Model analyser answers.

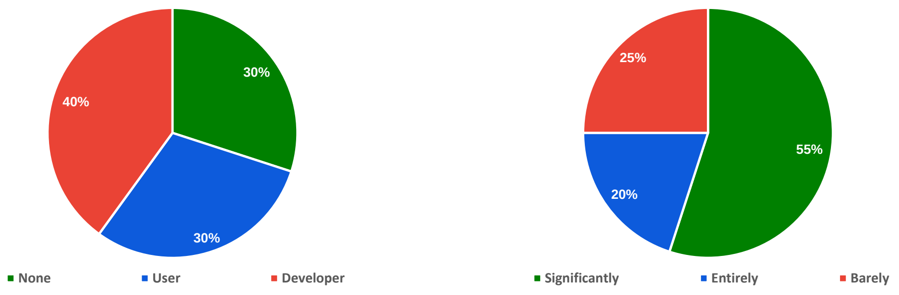


Figure 6: Expertise with integrated assessment model (left) and work relation respect to modelling tasks (right).

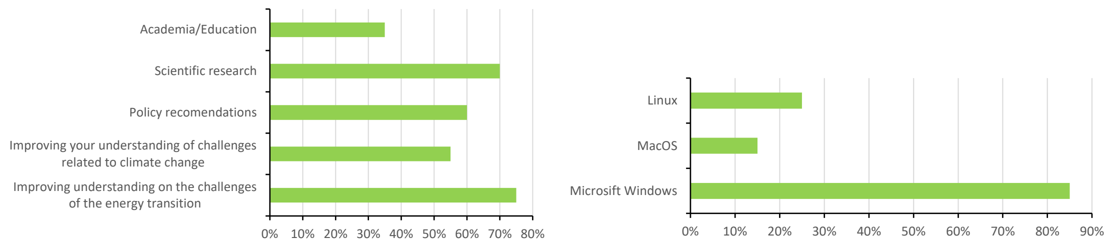


Figure 7: Use the model and results for (left) and selected operative system to use the app (right).

Considering the most relevant items to ply with in simulating the future, the users put the focus on social aspects, the GDP evolution, climate change impacts and adaptation policies, energy efficiency and the penetration of renewable technologies (Figure 8). The most relevant scenarios are the BAU and SSP while less than 50 parameters need to be required to evaluate the results (Figure 9). The collected answers reflect that the ideal time-frame for simulating is 2050 while the -csv format is the most adequate to visualize and download the results from each simulation (Figure 10).

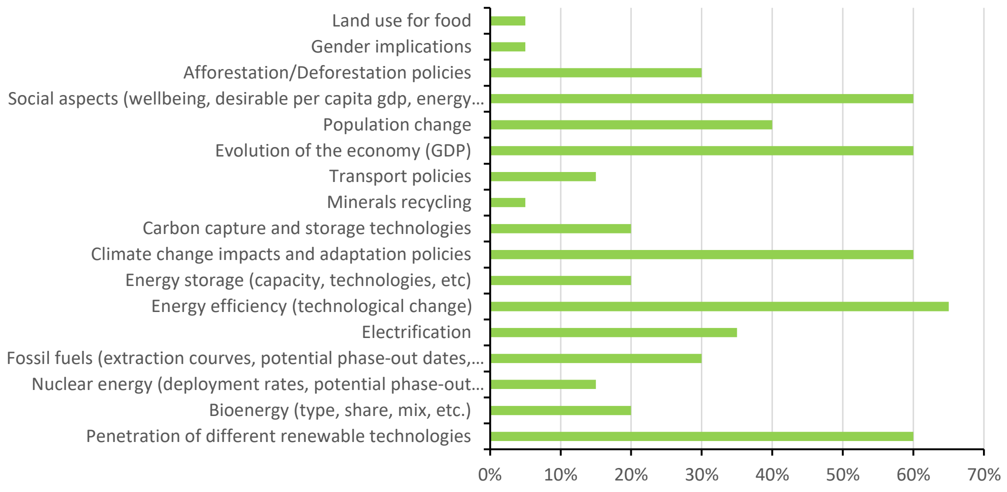


Figure 8: Most relevant items to play with when running simulations.

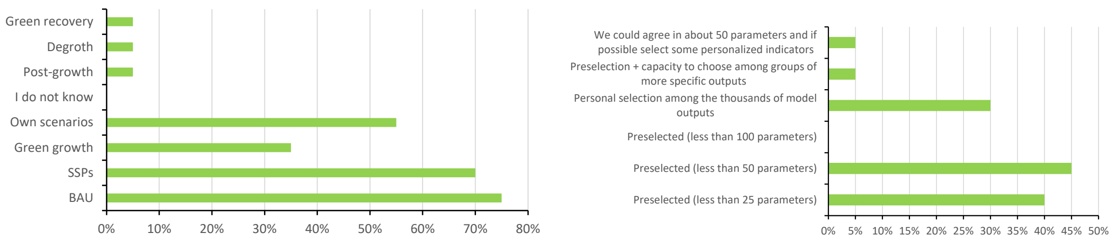


Figure 9: Pre-defined scenarios (left) and how to made the selection of model outputs (right).

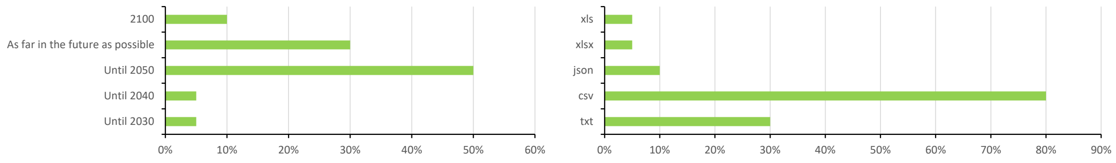


Figure 10: Time-frame to simulate the models (left) and format files for model inputs (right).

Respect to the visualization, graphs on the evolution of each variable is the most relevant option for the end-users (Figure 11). They point out that less than 10-20 minutes of simulation is required for the tool while running simulation in parallel could be an option if the model requires to much time in each simulation (Figure 12). Finally, the users consider the use of sliders, dropdowns, etc. joint external files as the most suitable option to parametrise the tool inputs while the tool need to have a documentation in .pdf as main option and that can be read in less than 15 minutes (Figure 13 y Figure 14)

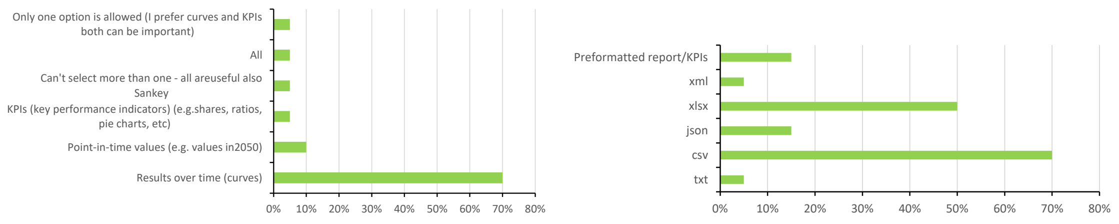


Figure 11: How to visualize model outputs (left) and format files for model outputs (right).

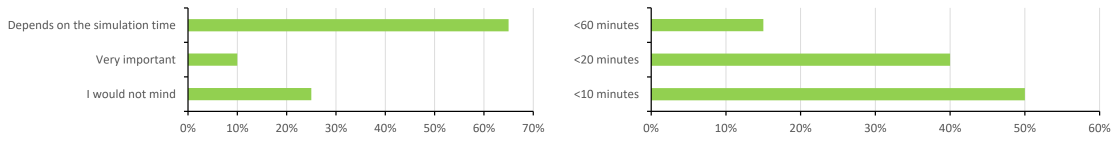


Figure 12: Interest in running simulations in parallel (left) and simulation time (right).

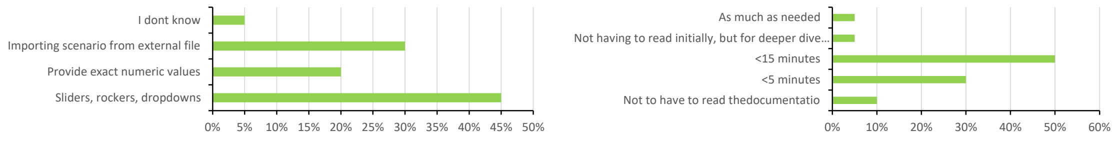


Figure 13: Method for model parametrization (left) and necessity of manuals and documentation (right).

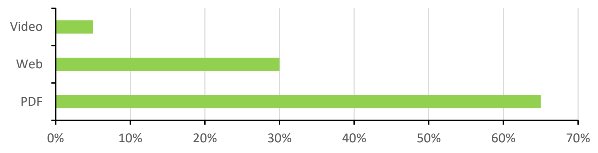


Figure 14: Most relevant format for manuals and documentation.

2.2.2. CROOSROADS II GAME

In this section, the results of the questionnaire to collect the requirements from the stakeholders that will help as the basis to guide the development of the Crossroads II Game is presented. Figure 15, presents the distribution of the sample among the different groups that are part of the project. The higher number of answers were collected from the consortium member followed by other stakeholders and SAB members. In general, answer are provided by professors o researcher who belong to the university community (Figure 16).

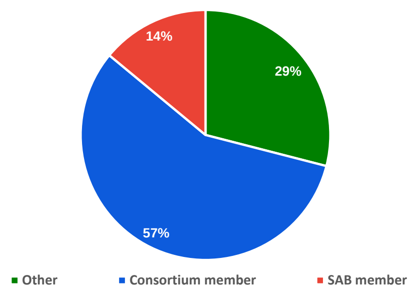


Figure 15: Role in LOCOMOTION project for Crossroads II Game answers.

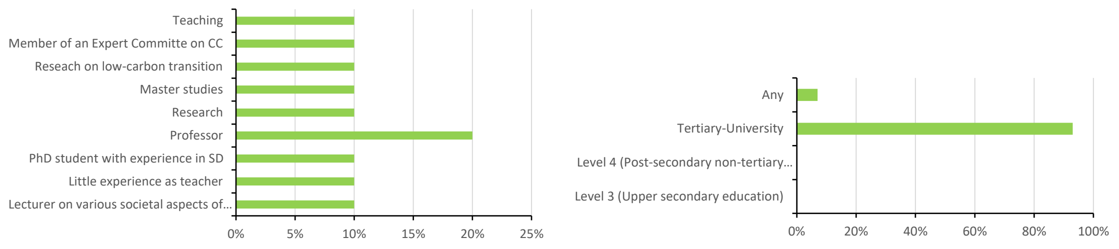


Figure 16: Relation of end-users with training and teaching on CC concerns (left) and the level of education the user has as student or educator (right).

End-users find very relevant the role of the game to contribute to improve knowledge on climate change and create awareness in their effect (Figure 17). They also point out that the game could be interesting to increase the relation between personal habits and climate change and also to create consensus about policies and regulation on climate change (Figure 18). The role of a moderator may be interesting for the use of the game in the educational community and outside of it, based on the opinions expressed in the responses received (Figure 19).

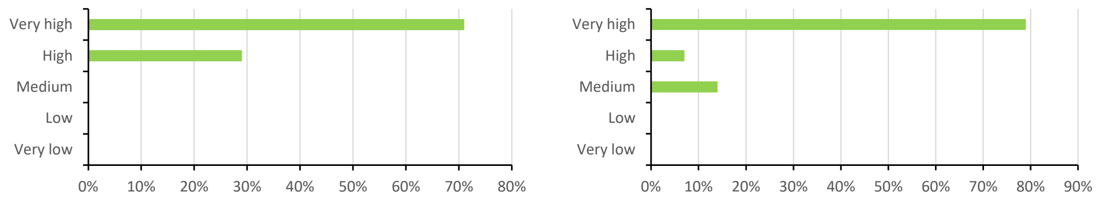


Figure 17: Opinion on how the game need to contribute to enrich knowledge on CC and its relation with energy and economy (left) and how it needs to contribute to create awareness on CC effects (right).

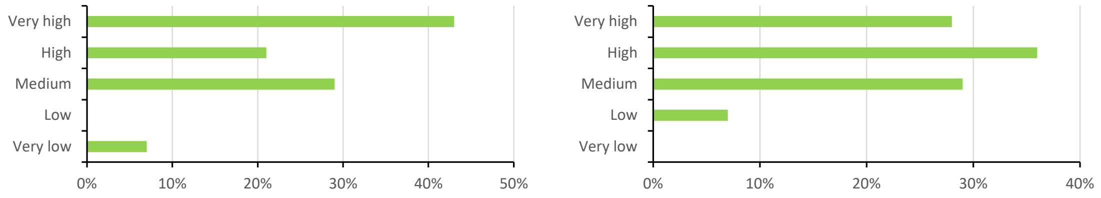


Figure 18: Opinion on how the game need to contribute to increase the relation of user's personal habits with CC impacts (left) and how it needs to encourage users to reach consensus in policies and regulations (right).

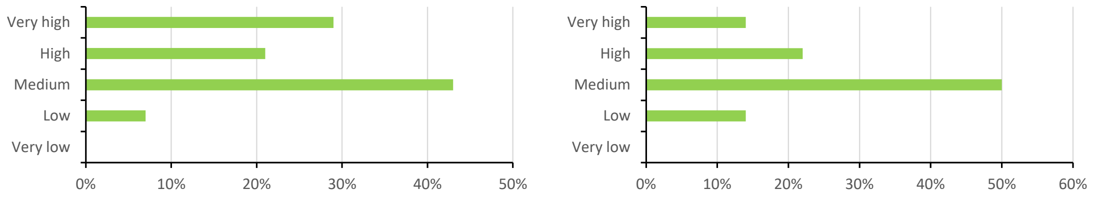


Figure 19: Opinion on how the role of a moderator is relevant in an education centre (left) and its relevance in a different context than an educational centre (right).

The future users think that it is very interesting that the game can be used to play alone as well as in groups (Figure 20) and that it should promote collaboration between the participants while the competitiveness between users should not be a differential criterion to be selected to design the game development interaction (Figure 21).

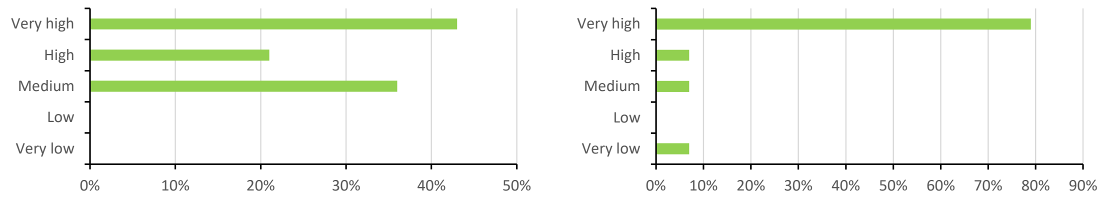


Figure 20: Opinion on the game for playing in teams (left) and playing alone (right).

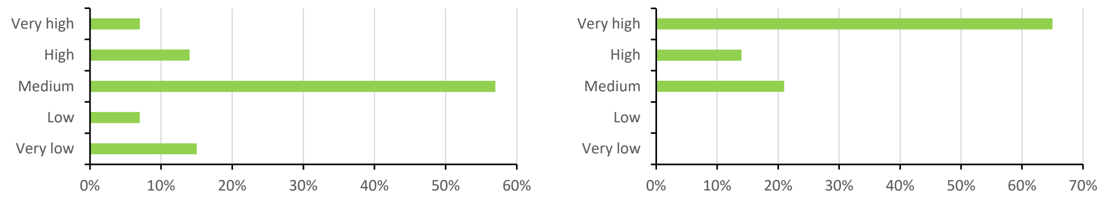


Figure 21: Opinion on the game for encourage competitiveness (left) and to promote collaboration between participants (right).

Finally, the users consider Windows app, web-based application and Linux app as the most suitable option to develop the Game (Figure 22). For them, the tool needs to have a game duration less than one hour and they would like to be able to play without having to read documentation or, if necessary, read documentation that requires less than 10 minutes (Figure 23)

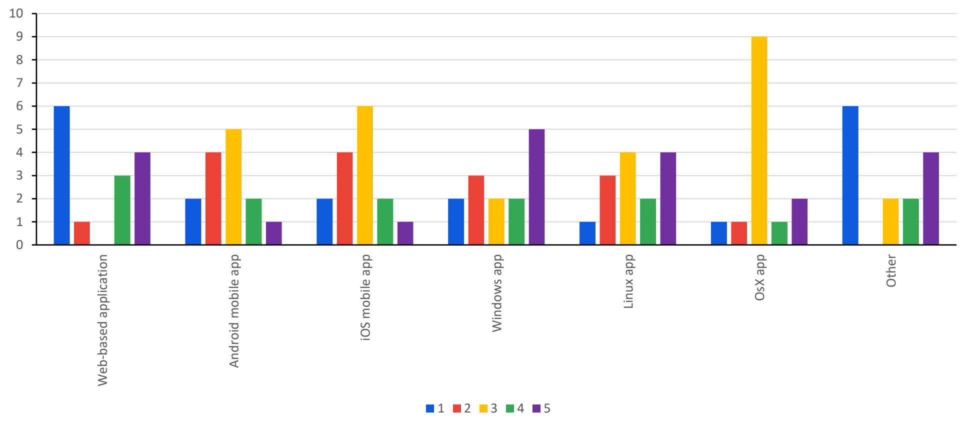


Figure 22: Opinion on the most suitable platform for Crossroads II Game.

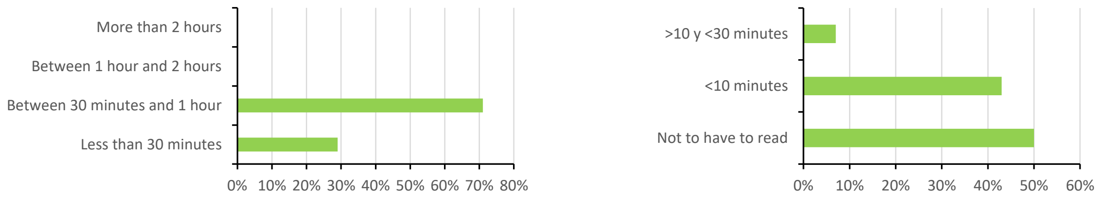


Figure 23: Opinion on the duration for the game session (left) and time spent on reading documentation (right).

2.2.3. MODEL EXPLORER

The following figures, present the results of the questionnaire to collect the requirements from the stakeholders that will help as the basis to guide the development of the Model Explorer. Figure 24, presents the distribution of the sample among the different groups that are part of the project. The higher number of answers were collected from the consortium member followed by other stakeholders with interest in LOCOMOTION project.

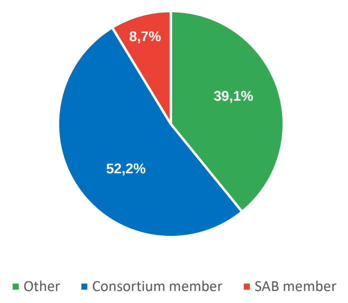


Figure 24: Role in LOCOMOTION project for Model Explorer answers.

According to the field of expertise, a valuable number of the collected answers comes from experts on energy, environment and climate change adaptation and mitigation (Figure 25). The put the focus on easy-to-use tools, with capabilities to visualize graphical data and export results (Figure 26). It is also relevant to highlight that comparison of results between regions could be relevant while the comparison of your own scenario and the BAU is crucial for the users. Considering the input data Figure 27), it is necessary to reflect that users prioritize policy to change energy consumption, renewable share, electrification and economic growth among others.

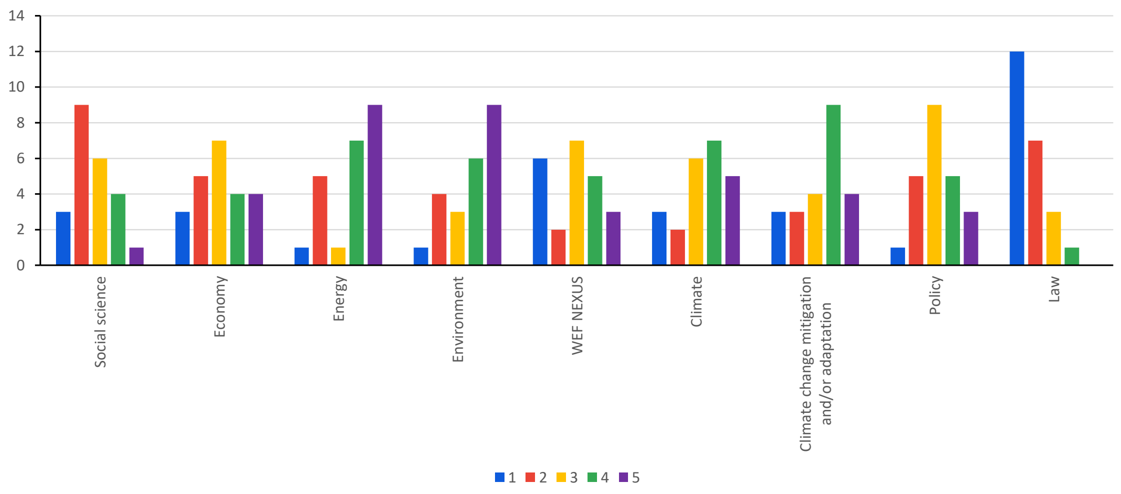


Figure 25: Expertise by field for Model Explorer answers.

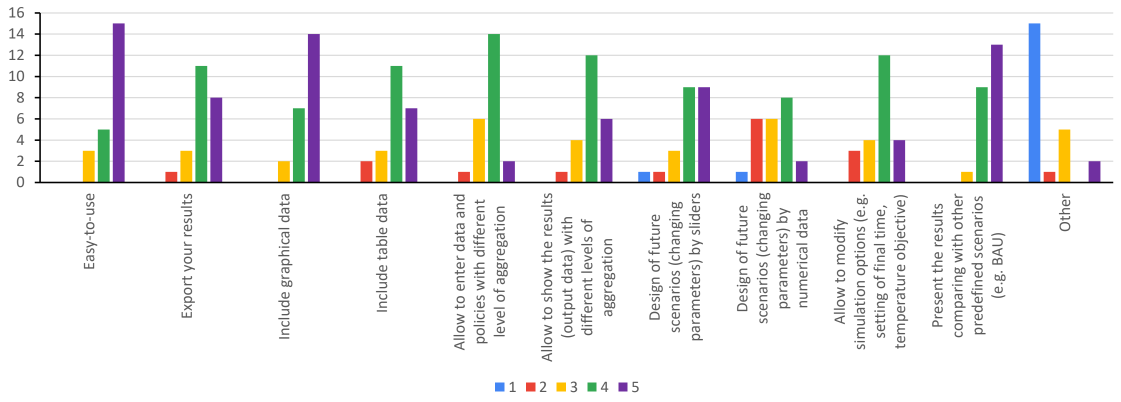


Figure 26: Criteria to define a good web-based tool.

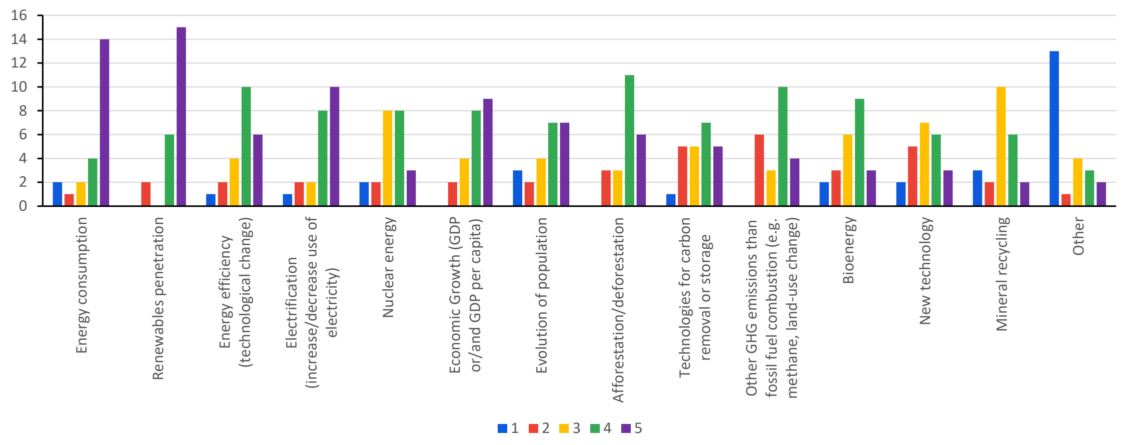


Figure 27: Type of input data to be able to modify.

If we ask about the most interesting scenarios to be considered, the end-users raised the BAU and SSP scenarios as the most interesting followed by the custom scenario developed by the users (Figure 28). Related with the hypothesis, they consider climate change impacts as the most relevant hypothesis to be tested with the use of the tool in the scenario implementation. The COVID-19 scenario is very relevant for the 21% of the end-users, while the 30% of them consider relevant to include policies related with crisis similar as COVID-19 (Figure 29).

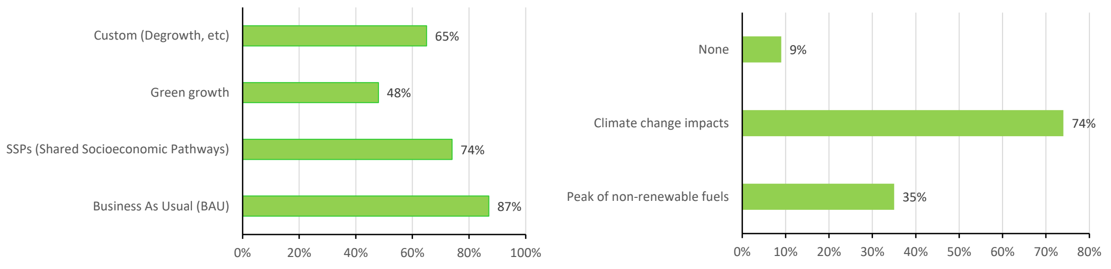


Figure 28: Predefined scenarios to be considered (left) and simulation hypothesis (right).

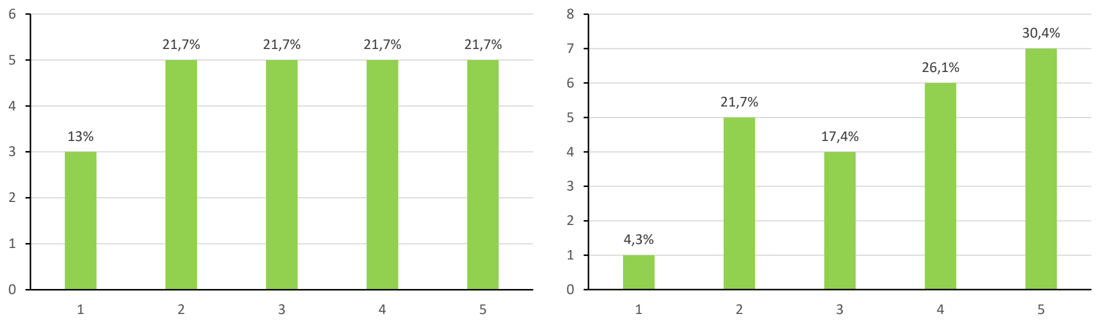


Figure 29: Relevance of COVID-19 scenario (left) and relevance of COVID-19 policies (right).

Finally, it is necessary to highlight the most relevant outputs when analyzing the impacts of the implemented policies. The future users of the tools consider the change in temperature, CO_2 emissions and other greenhouse gases as the most relevant variables, as well as the demand and consumption of energy (Figure 30). To conclude, almost 74% of those surveyed have no previous experience in the use of web tools similar to the one LOCOMOTION will develop (Figure 31).

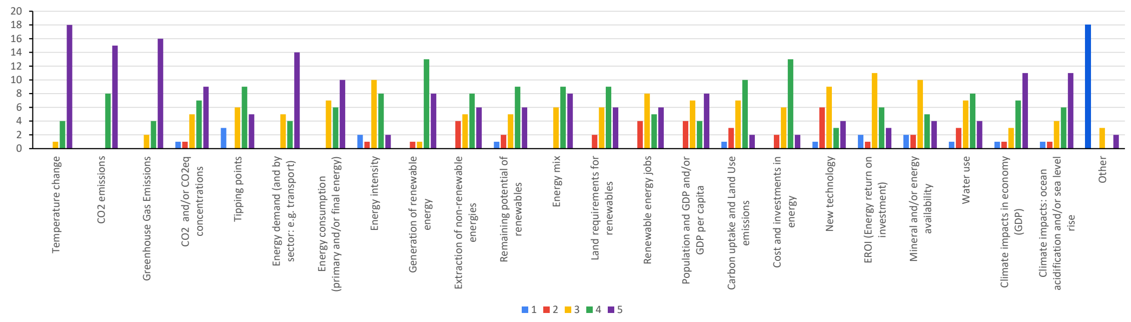


Figure 30: Expected outputs to visualize.

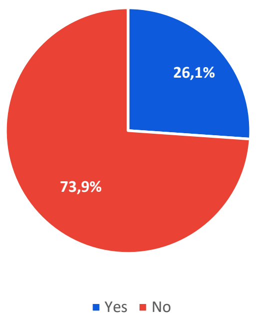


Figure 31: Previous experience with web-based tools.

3. MODEL ANALYSER

3.1. IDENTIFIED REQUIREMENTS

The Model Analyzer is a desktop application, which allows configuring scenarios and running them using the WILIAM model. The expected audience for this app is experts on the energy transition and policy makers. The Model Analyzer was originally defined as a standalone (off-line) tool, that users may download, install and run in their computers.

This approach is successfully used by C-ROADS (https://www.climateinteractive.org/tools/c-roads/), which provides both offline and online alternatives. However, the computational cost of the C-ROADS and WILIAM models are not comparable. Indeed, while the computation time of the first lays under 1 second with modern desktop computers, simulations run with WILIAM were estimated to take over 120 minutes (with comparable hardware). This extra hardware requirement may limit the access to this tool to potential lower-income users.

In addition, the aforementioned approach would require the installation of proprietary software (though free of charge) on the user's machines (Vensim® Model Reader). Also, Vensim® Model Reader is only available for the Windows and Mac operative systems, and does not run natively on the GNU/Linux platform, which once again may restrict accessibility to the tool. The need to give GNU/Linux users the possibility to run the model natively on their machines was backed by the results of the surveys performed in WP11.

Another inconvenience of the standalone off-line approach was that, in the early stages of the WILIAM model development, newer versions were expected to be released in a relatively high frequency. If we add to that the potential software updates to the Vensim® Model Reader and the other pieces of software required to make all the package work, this would result in several reinstallations of the software in a short period of time, affecting the final user experience.

Considering the previous issues, which were fully discussed with the partners involved in the task over several successive meetings, it was decided to switch to a client-server architecture. With this new approach, the product's functionality is the same, but the proprietary software is installed on a remote server, which also takes the computational cost associated with the simulations. Once the simulation has ended, the user can download the simulation results on their machine, which allows them to visualize and compare results from several simulations offline.

This new approach requires the user to have Internet access, but only at the time of launching the simulations and collecting the results. While the simulation is running on the server, the user may choose to disconnect from the Internet without it causing the simulation to be interrupted. When their simulations are finished, the user is notified. Our best guess is that the access to modern hardware (and proprietary operative systems) will be more restricting to potential users than the intermittent access to the Internet. Although in many cases having a right does not necessarily imply being able to exercise it, Internet access was declared a human right by the UN in 2016.

The main advantages of the suggested approach are:

- Reduces the user hardware requirements and the simulation time
- No need to install proprietary software on the user machine
- The software will run natively in more operative systems (e.g. GNU/Linux)
- The user always runs on the latest version of the model (automatic and transparent model updates)
- Allows the users to run parallel batch simulations
- Opens the possibility to collect anonymous data to improve the model/user experience and to perform sophisticated data analysis from the user's simulation results
- The whole infrastructure can also be used for research purposes

The Model Analyzer has two predefined scenarios, one with full configuration options (in terms of variables dimensions and number of available parameters) and another one with simplified configuration options. The model that is run in the background is the same.

3.2. MOCKUPS

The initial mockup of the Model Analyzer was created using Microsoft PowerPoint. A separate slide was created for each of the main pages of the app, as shown in Figure 32 to Figure 41.

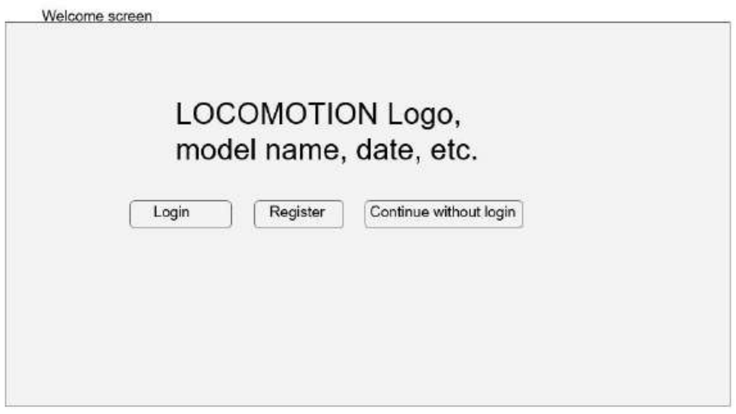


Figure 32: Welcome page.

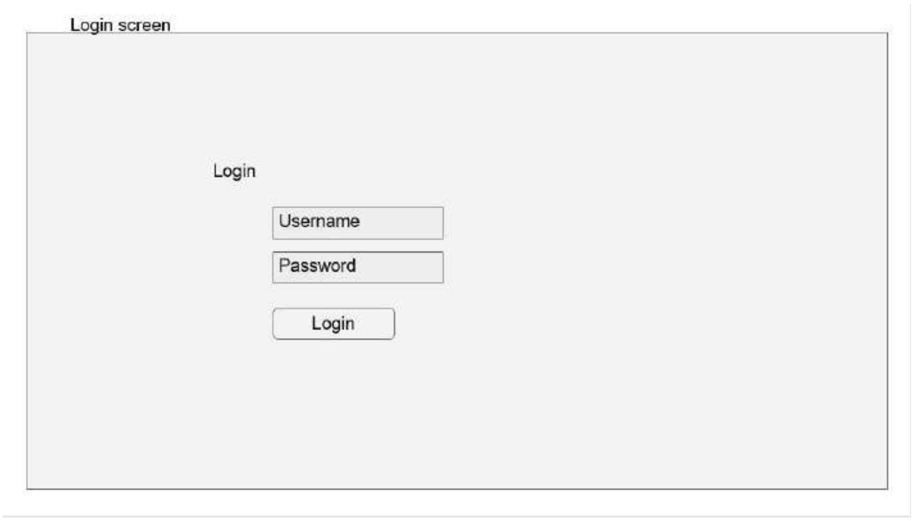
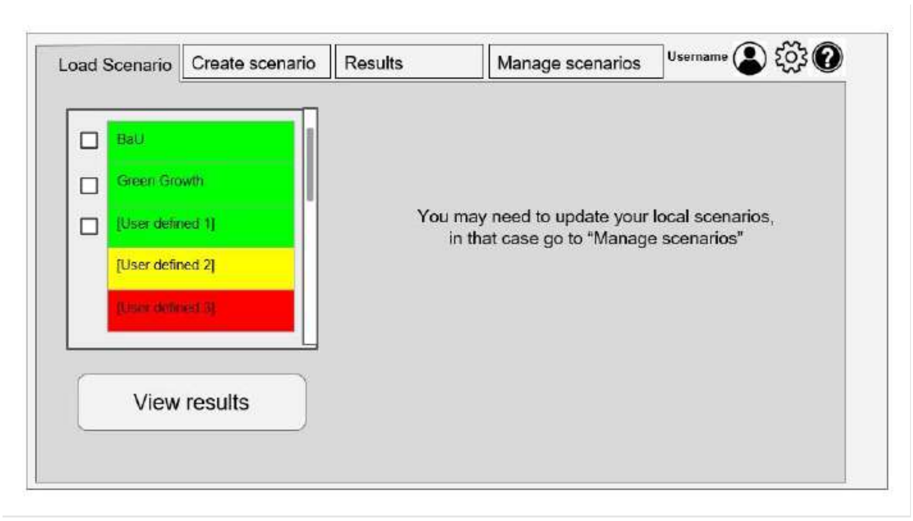
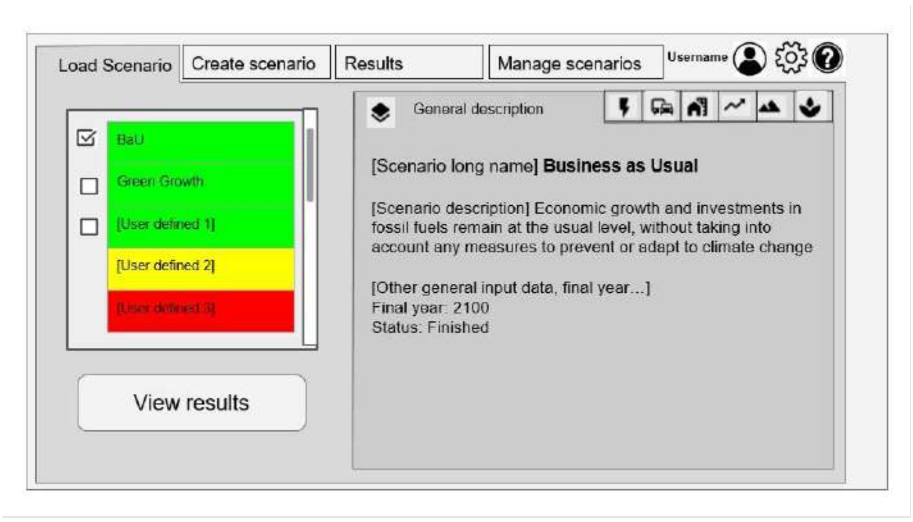


Figure 33: Log in page.



You may need to update your local scenarios, in that case go to "Manage scenarios"

Figure 34: Load scenario page (later named Scenarios): main view.

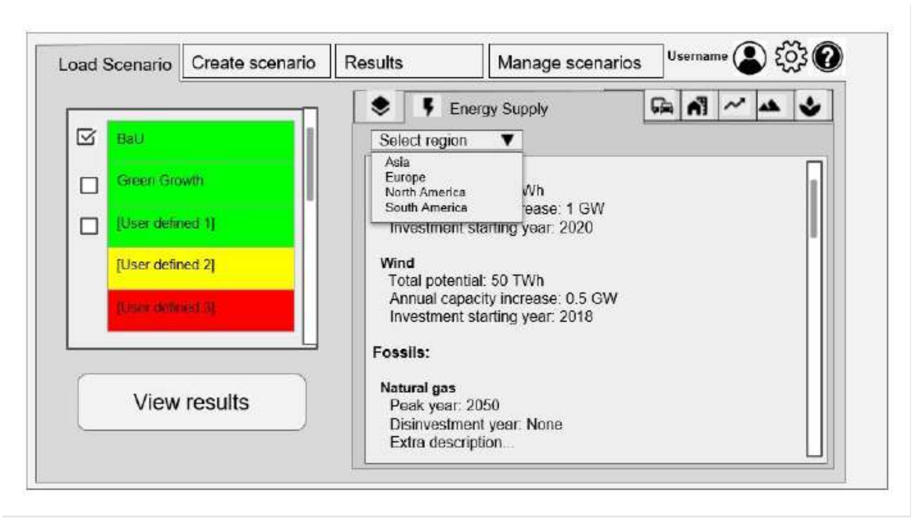


[Scenario long name] **Business as Usual**

[Scenario description] Economic growth and investments in fossil fuels remain at the usual level, without taking into account any measures to prevent or adapt to climate change

[Other general input data, final year...]
Final year: 2100
Status: Finished

Figure 35: Load scenario page (later named Scenarios): display scenario description.


Load Scenario | Create scenario | Results | Manage scenarios | Username

☑ BaU
☐ Green Growth
☐ [User defined 1]
[User defined 2]
[User defined 3]

View results

Energy Supply

Select region ▼
Asia
Europe
North America
South America

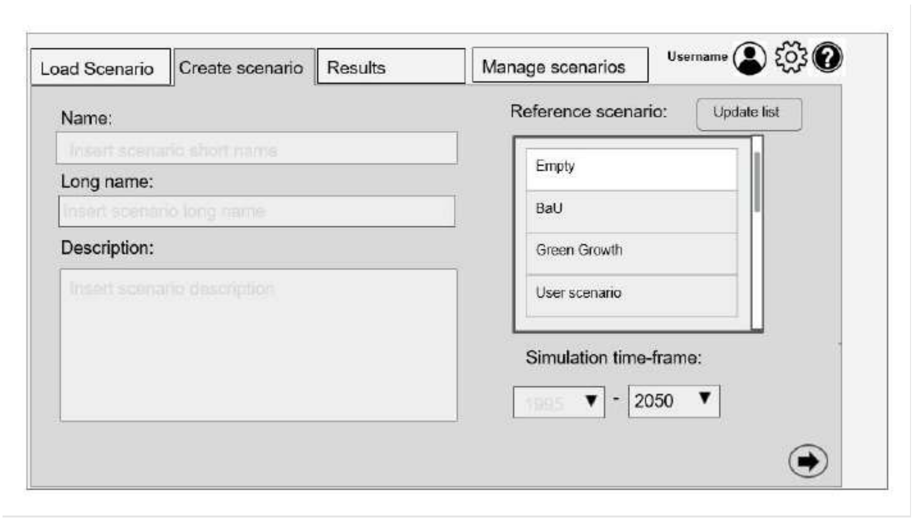
Wh
ease: 1 GW
Investment starting year: 2020

Wind
Total potential: 50 TWh
Annual capacity increase: 0.5 GW
Investment starting year: 2018

Fossils:

Natural gas
Peak year: 2050
Disinvestment year: None
Extra description...

Figure 36: Load scenario page (later named Scenarios): display scenario parameter values.

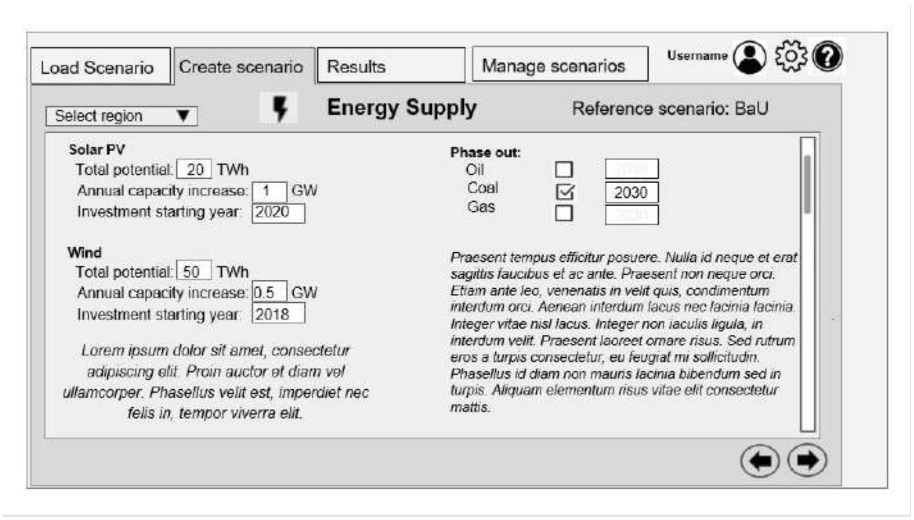

Load Scenario | Create scenario | Results | Manage scenarios | Username

Name:
Insert scenario short name
Long name:
Insert scenario long name
Description:
Insert scenario description

Reference scenario: Update list
Empty
BaU
Green Growth
User scenario

Simulation time-frame:
1995 ▼ - 2050 ▼

Figure 37: Create scenario page (later named New Scenario): configuring the name, description, reference scenario and simulation time-frame.


Load Scenario | Create scenario | Results | Manage scenarios | Username

Select region ▼ | **Energy Supply** | Reference scenario: BaU

Solar PV
Total potential: 20 TWh
Annual capacity increase: 1 GW
Investment starting year: 2020

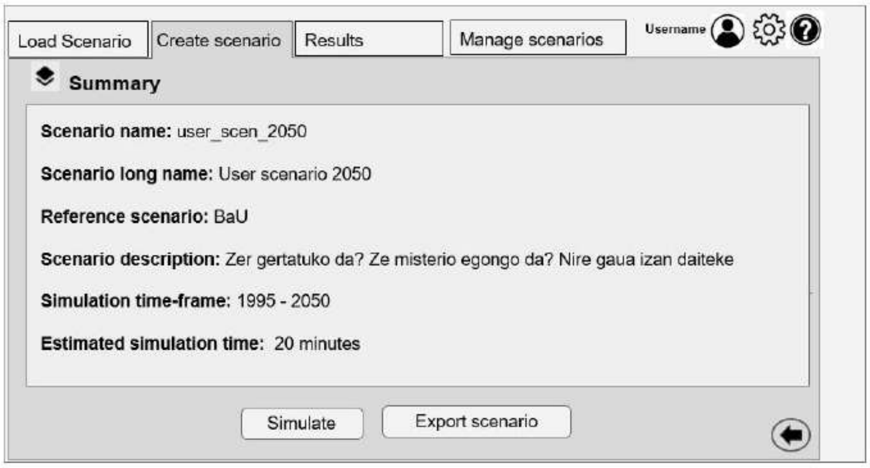
Wind
Total potential: 50 TWh
Annual capacity increase: 0.5 GW
Investment starting year: 2018

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin auctor et diam vel ullamcorper. Phasellus velit est, imperdiet nec felis in, tempor viverra elit.

Phase out:		
Oil	☐	2030
Coal	☑	2030
Gas	☐	2030

Praesent tempus efficitur posuere. Nulla id neque et erat sagittis faucibus et ac ante. Praesent non neque orci. Etiam ante leo, venenatis in velit quis, condimentum interdum orci. Aenean interdum lacus nec lacinia lacinia. Integer vitae nisl lacus. Integer non iaculis ligula, in interdum velit. Praesent laoreet ornare risus. Sed rutrum eros a turpis consectetur, eu feugiat mi sollicitudin. Phasellus id diam non mauris lacinia bibendum sed in turpis. Aliquam elementum risus vitae elit consectetur mattis.

Figure 38: Create scenario page (later named New Scenario): configuring scenario parameters.



Scenario name: user_scen_2050

Scenario long name: User scenario 2050

Reference scenario: BaU

Scenario description: Zer gertatuko da? Ze misterio egongo da? Nire gaua izan daiteke

Simulation time-frame: 1995 - 2050

Estimated simulation time: 20 minutes

Figure 39: Create scenario page (later named New Scenario): summary of the newly created scenario.

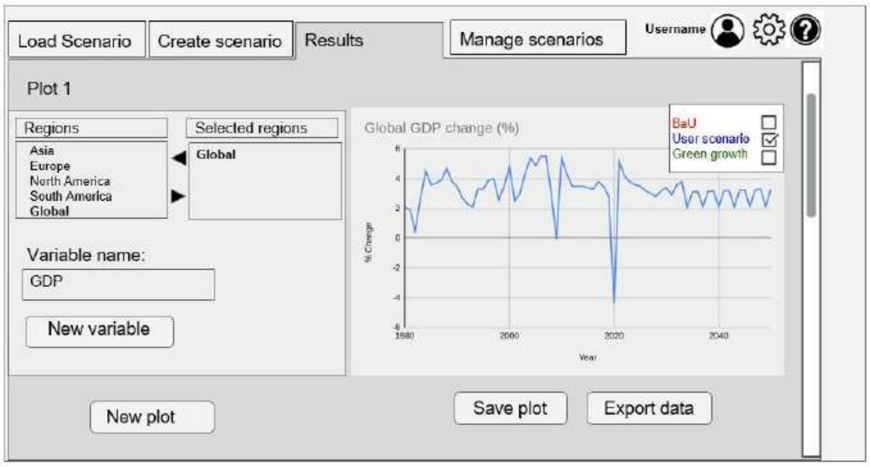


Figure 40: Results page (later named Plot Results).

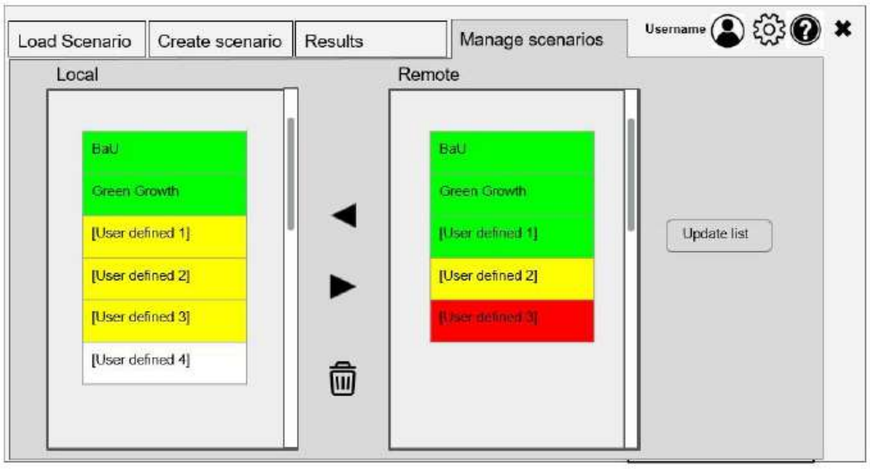


Figure 41: Manage Scenarios page: colour code to visualize the state of the simulations in the server and locally. Possibility to simulate scenarios on the server, download scenarios from the server, or delete scenarios from remote or local.

3.3. GUI DESIGN

Based on the generated mockups, the design was created using the InVision platform (https://www.invisionapp.com/). This task was carried out in collaboration with Ondeuev, based on the Graphical Identity Manual of the project, with the objective of maintaining a design uniformity across the three apps. The final design of the interfaces in presented in Figure 42 to Figure 61.

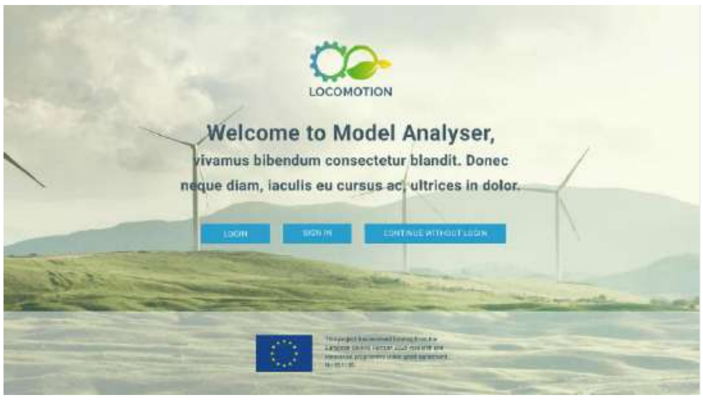

Welcome to Model Analyser,
vivamus bibendum consectetur blandit. Donec neque diam, iaculis eu cursus ac, ultrices in dolor.

Figure 42: Design of the welcome page.

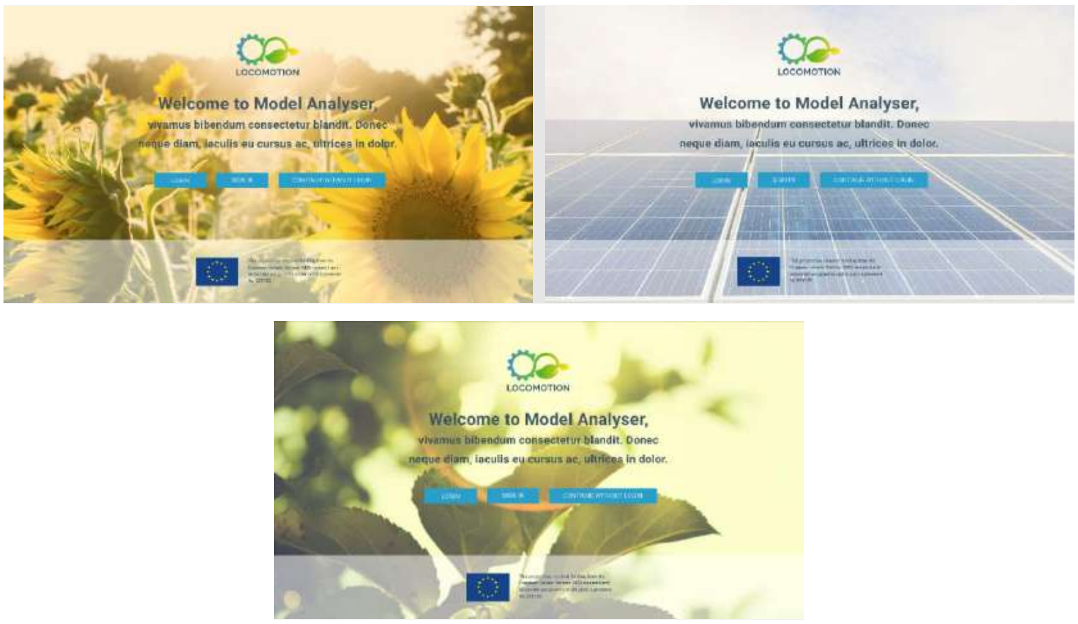

Welcome to Model Analyser,
vivamus bibendum consectetur blandit. Donec neque diam, iaculis eu cursus ac, ultrices in dolor.

Figure 43: Design of the welcome page: alternative background images.

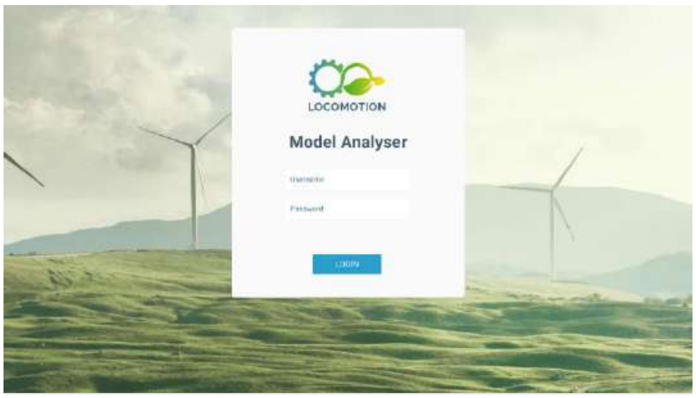


Figure 44: Design of the log in page.

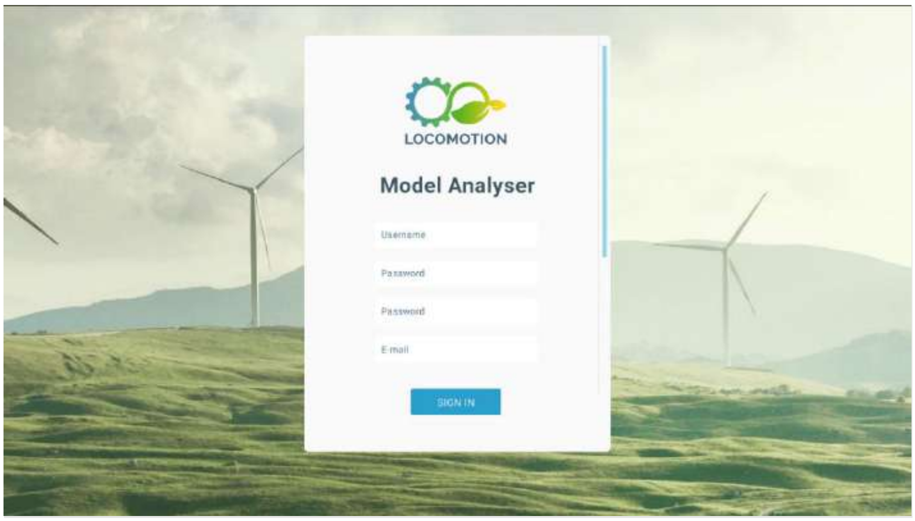
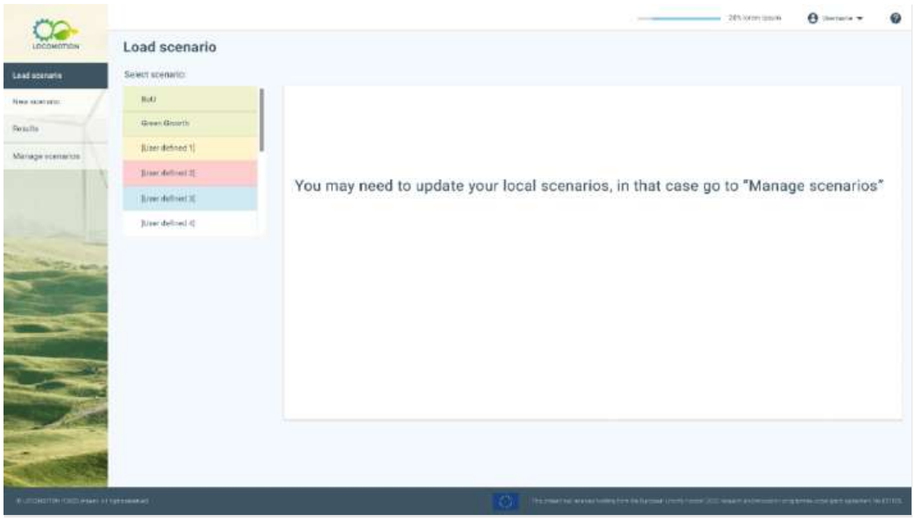
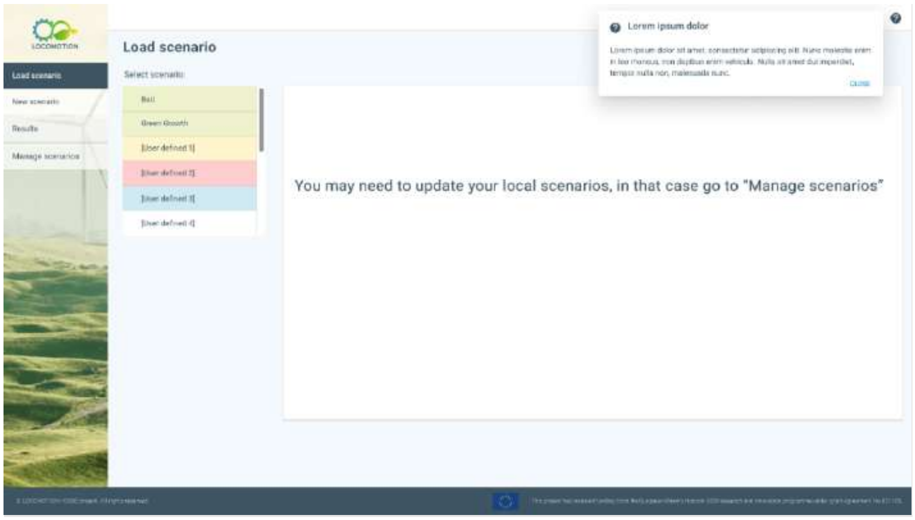


Figure 45: Design of the sign in page.



You may need to update your local scenarios, in that case go to "Manage scenarios"

Figure 46: Design of the Load scenario page (later named Scenarios): main page.

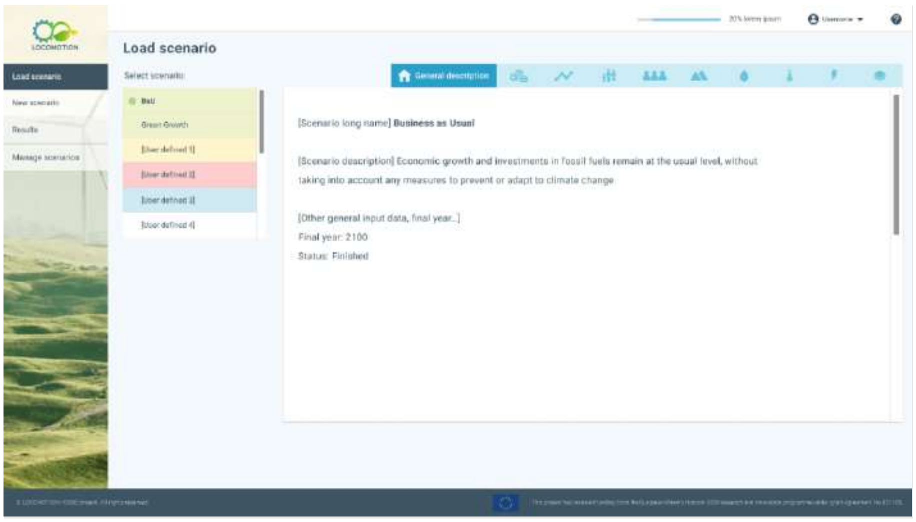


Lorem ipsum dolor

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc molestie enim in leo rhoncus, non dapibus enim vehicula. Nulla sit amet dui imperdiet, tempor nulla non, malesuada nunc.

You may need to update your local scenarios, in that case go to "Manage scenarios"

Figure 47: Design of the Load scenario page (later named Scenarios): information message.



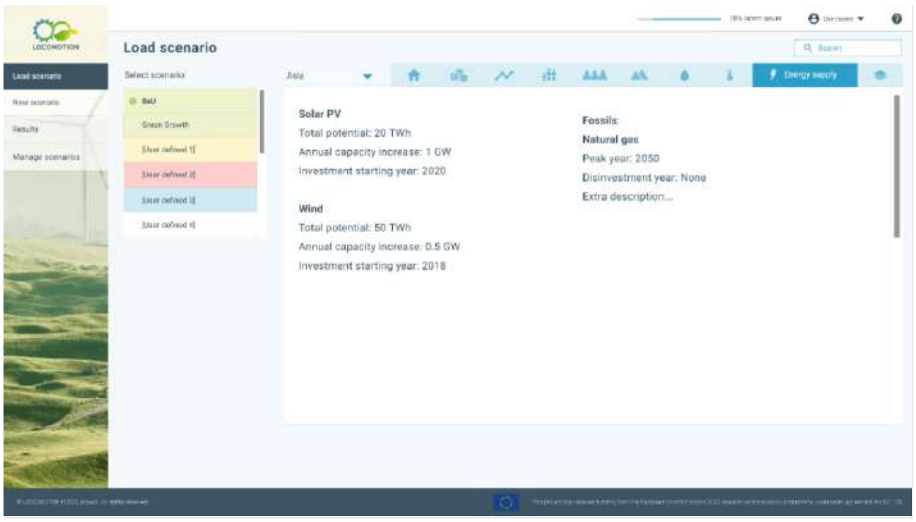
Load scenario

[Scenario long name] **Business as Usual**

[Scenario description] Economic growth and investments in fossil fuels remain at the usual level, without taking into account any measures to prevent or adapt to climate change

[Other general input data, final year...]
Final year: 2100
Status: Finished

Figure 48: Design of the Load scenario page (later named Scenarios): summary of the newly created scenario.



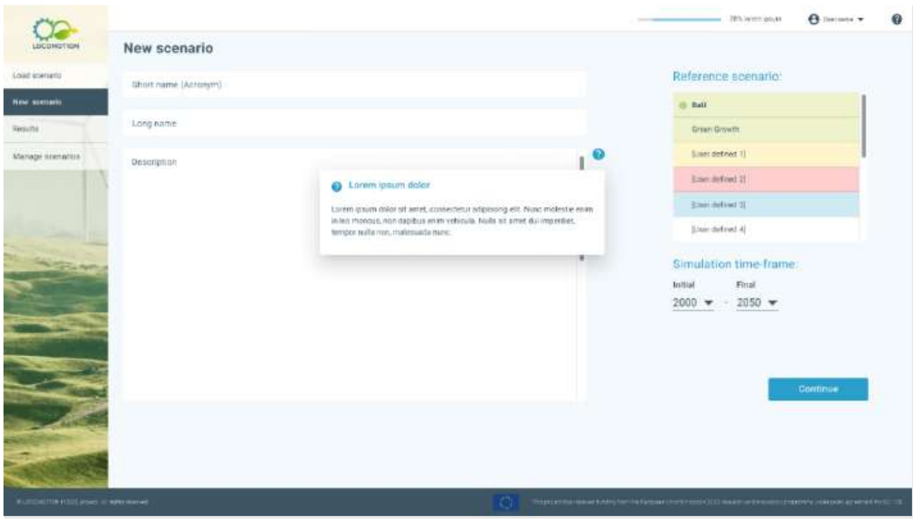
Load scenario

Solar PV
Total potential: 20 TWh
Annual capacity increase: 1 GW
Investment starting year: 2020

Wind
Total potential: 50 TWh
Annual capacity increase: 0.5 GW
Investment starting year: 2018

Fossils
Natural gas
Peak year: 2050
Disinvestment year: None
Extra description...

Figure 49: Design of the Load scenario page (later named Scenarios): visualization of scenario parameters and values.

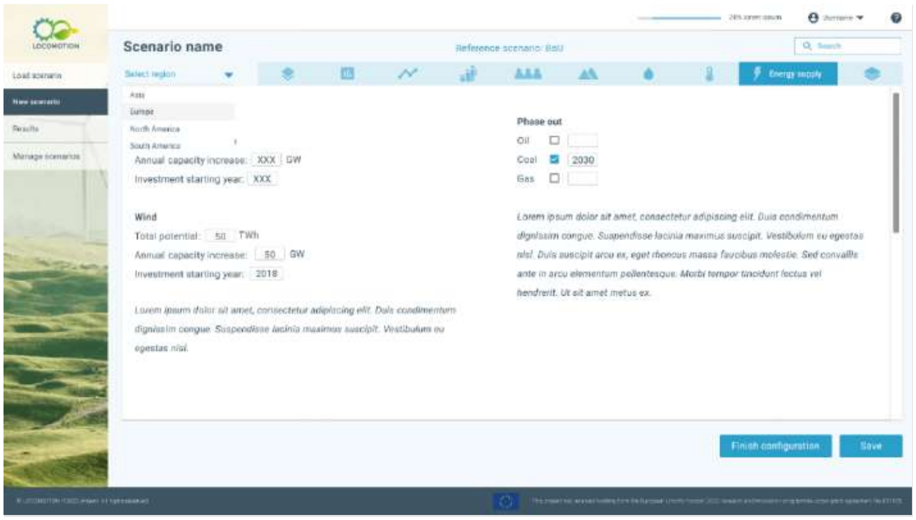


New scenario

Lorem ipsum dolor

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc molestie enim in leo rhoncus, non dapibus enim vehicula. Nulla sit amet dui imperdiet, tempor nulla non, malesuada nunc.

Figure 50: Design of the New Scenario page.



Annual capacity increase: XXX GW
Investment starting year: XXX

Wind
Total potential: 50 TWh
Annual capacity increase: 50 GW
Investment starting year: 2018

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis condimentum dignissim congue. Suspendisse lacinia maximus suscipit. Vestibulum eu egestas nisl.

Phase out
Oil ☐
Coal ☑ 2030
Gas ☐

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis condimentum dignissim congue. Suspendisse lacinia maximus suscipit. Vestibulum eu egestas nisl. Duis suscipit arcu ex, eget rhoncus massa faucibus molestie. Sed convallis ante in arcu elementum pellentesque. Morbi tempor tincidunt fectus vel hendrerit. Ut sit amet metus ex.

Figure 51: Design of the New scenario page: editing scenario parameter values.

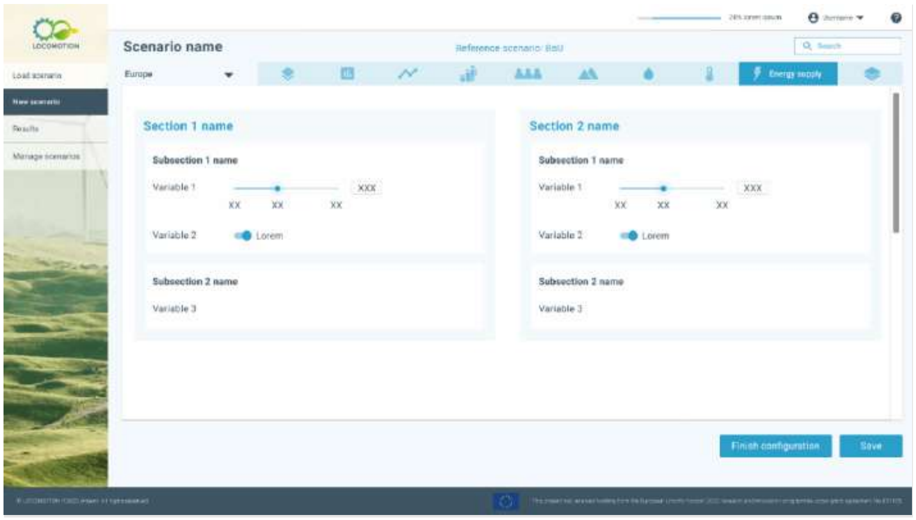
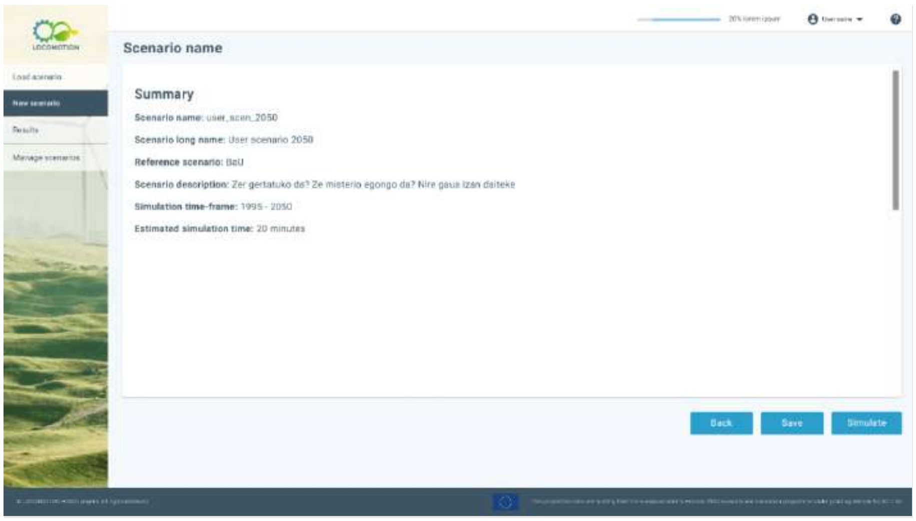


Figure 52: Design of the New scenario page: widgets.



Summary

Scenario name: user_scen_2050
Scenario long name: User scenario 2050
Reference scenario: BaU
Scenario description: Zer gertatuko da? Ze misterio egongo da? Nire gaua izan daiteke
Simulation time-frame: 1995 - 2050
Estimated simulation time: 20 minutes

Figure 53: Design of the New scenario page: scenario summary.

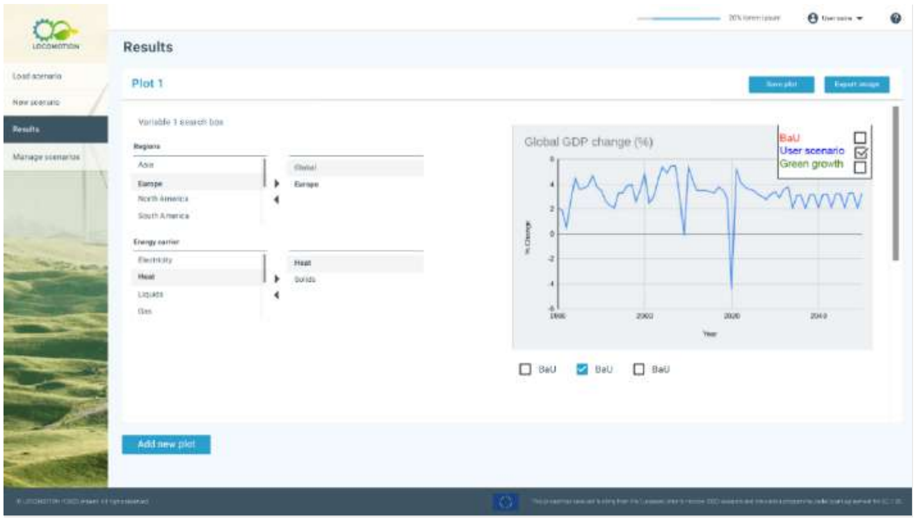


Figure 54: Design of the Results page (later named Plot Results).

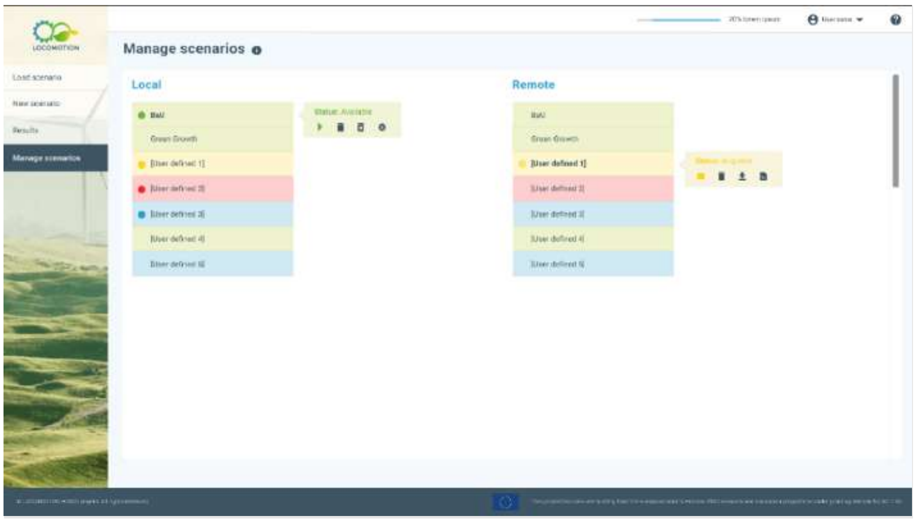


Figure 55: Design of the Manage Scenarios page.

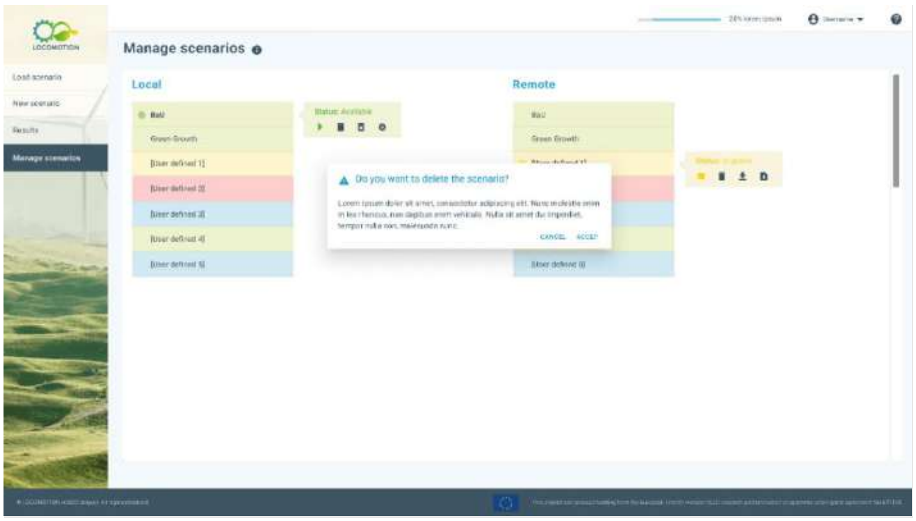
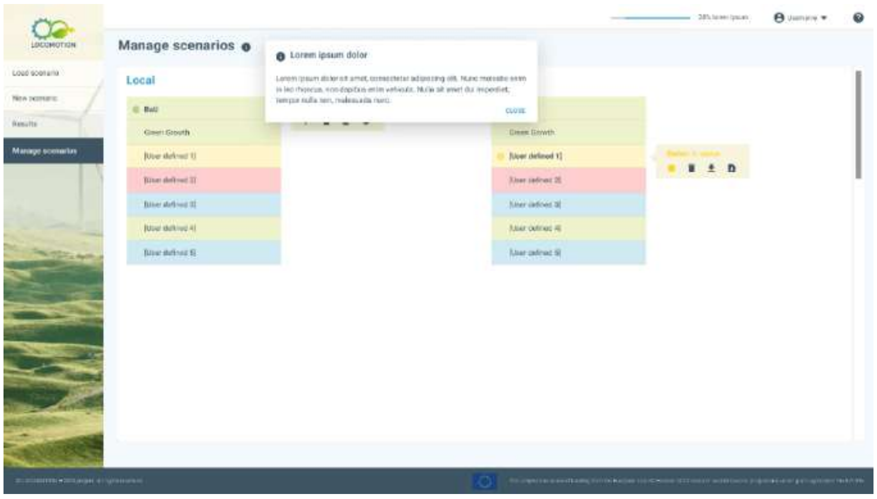


Figure 56: Design of the Manage Scenarios page: warning message.



Lorem ipsum dolor

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc molestie enim in leo rhoncus, non dapibus enim vehicula. Nulla sit amet dui imperdiet, tempus nulla non, malesuada nunc.

Figure 57: Design of the Manage Scenarios page: information message.

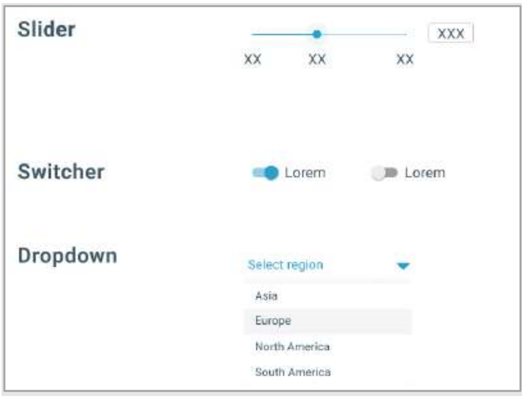
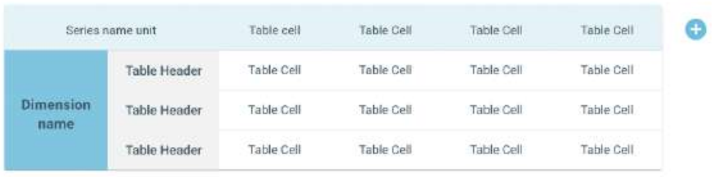
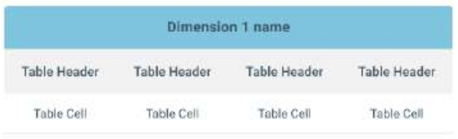


Figure 58: Design of widgets.



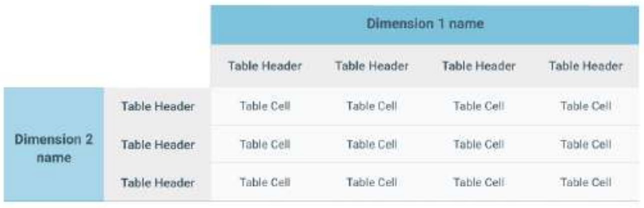
Series name unit		Table cell	Table Cell	Table Cell	Table Cell
Dimension name	Table Header	Table Cell	Table Cell	Table Cell	Table Cell
	Table Header	Table Cell	Table Cell	Table Cell	Table Cell
	Table Header	Table Cell	Table Cell	Table Cell	Table Cell

Figure 59: Design of tables for Vensim® Lookups and Data types.



Dimension 1 name			
Table Header	Table Header	Table Header	Table Header
Table Cell	Table Cell	Table Cell	Table Cell

Figure 60: Design for vectors.



		Dimension 1 name			
		Table Header	Table Header	Table Header	Table Header
Dimension 2 name	Table Header	Table Cell	Table Cell	Table Cell	Table Cell
	Table Header	Table Cell	Table Cell	Table Cell	Table Cell
	Table Header	Table Cell	Table Cell	Table Cell	Table Cell

Figure 61: Design for multi-dimensional matrices.

3.4. SELECTED TECHNOLOGIES

The backend of the Model Analyzer is entirely written in Python. It includes an API REST built using the Guillotina library (https://guillotina.readthedocs.io/en/latest/), NATS for data exchange and the asyncIO library (https://docs.python.org/3/library/asyncio.html) to process NATS messages and generate individual subprocesses. Vensim® DSS 9.3.4 is the simulation engine used to run the simulations in subprocesses. Finally, a Postgress database is used to store user's authentication information and Redis is used for caching. Docker is used to containerize parts of the backend.

On the other hand, the desktop app is built using the Electron framework (https://www.electronjs.org/) combined with React (https://reactjs.org/) and with Material-UI components (https://mui.com/).

3.5. ARCHITECTURE

In the backend, there is a Windows server and a GNU/Linux server, which are hosted in the premises of the *Parque Científico de la Universidad de Valladolid*.

Services that run on the Linux server, include:

- The Guillotina REST API, NATS server (for communication with the Windows server), Postgress database and Redis, all in a docker container (Dockerfile)
- SMTP server to send emails to users
- Docker secrets (https://docs.docker.com/engine/swarm/secrets/)

Services that run on the Windows server:

- NATS client to communicate with the Linux server
- Workers: the workers script is written in Python and an OS native library. Workers get messages from the NATS JStream, they process them and open the Vensim® software to compute the model with the given payload coming from the message.
- Samba server so that the GNU/Linux server can access simulation results (E:\VensimData\results) and the payloads (E:\Models\inputs).

The interface of the app is presented in Section 0 in this document. The desktop app communicates directly with the Guillotina API Linux server. The app can be installed using the packages that are built for Windows (exe), Mac (dmg) and Linux (.deb and .rpm) (see Section 0).

3.6. TOOL IMPLEMENTATION

EBAVS/ worked in close collaboration with Ondeuev to build the layout of the application, with supervision from CREAF.

CREAF and ISKRA collaborated on the development of the backend. ISKRA worked primarily on the software architecture and CREAF on all the code required to create the scenarios (using data from the WILIAM model and metadata from the Data Client) and to communicate with Vensim® software.

The tool was developed and tested directly on the hardware that would be used for production.

The development was tracked using the Shortcut project management software (https://shortcut.com/).

3.7. QUALITY ASSURANCE

The whole backend workflow is tested with Pytest unit tests.

Each successive release was thoroughly reviewed and tested by CREAF. Both toy models and initial releases of the WILIAM model were used for that purpose.

When development was over, a production-ready release was shared with all LOCOMOTION partners and their feedback was incorporated for the final Model Analyzer release.

3.8. DEVELOPER GUIDE AND DEVELOPMENT

3.8.1. CONFIGURING SCENARIOS IN THE DATA CLIENT

The two available scenarios in the Model Analyzer (Full and Simplified) are configured from the LOCOMOTION Data Client. From there, administrators can decide which scenario parameters and which WILIAM output variables will be displayed in each of the two scenarios in the Desktop app. Administrators can also provide the required metadata in the Data Client so that all parameters are correctly displayed in the interface (clean variable names, dimensions, units and descriptions, sections and subsections, etc.). In the following lines, we show how all this is configured.

A prerequisite for Symbols to be shown in the Model Analyzer Desktop app is that its type is set to either *Scenario Parameter* or *Variable*. To do so:

1. Log in the Data Client.
2. Click on the Symbols/Manage symbols menu (Figure 62).

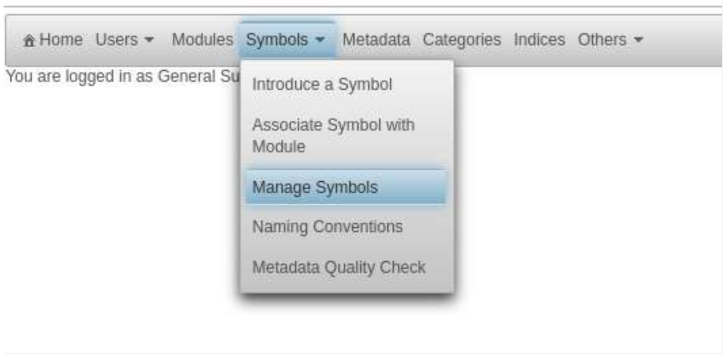


Figure 62: Data Client Symbols menu.

3. Click on the Symbol to be edited to make visible in the Desktop app, and then click on the Edit button, at the bottom of the screen (Figure 63).

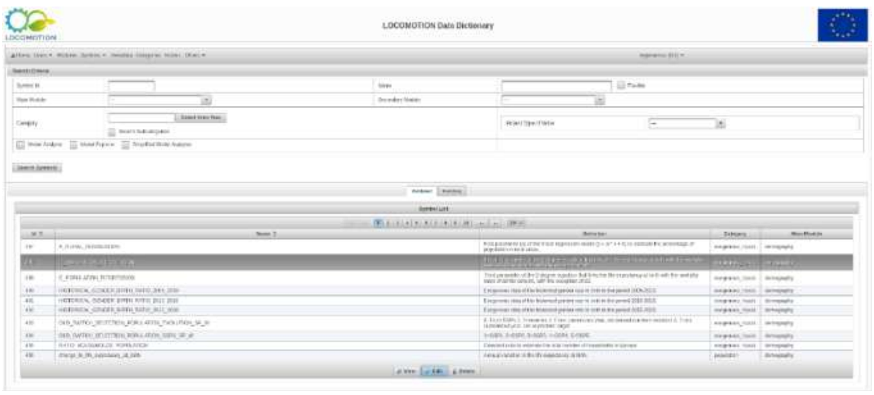
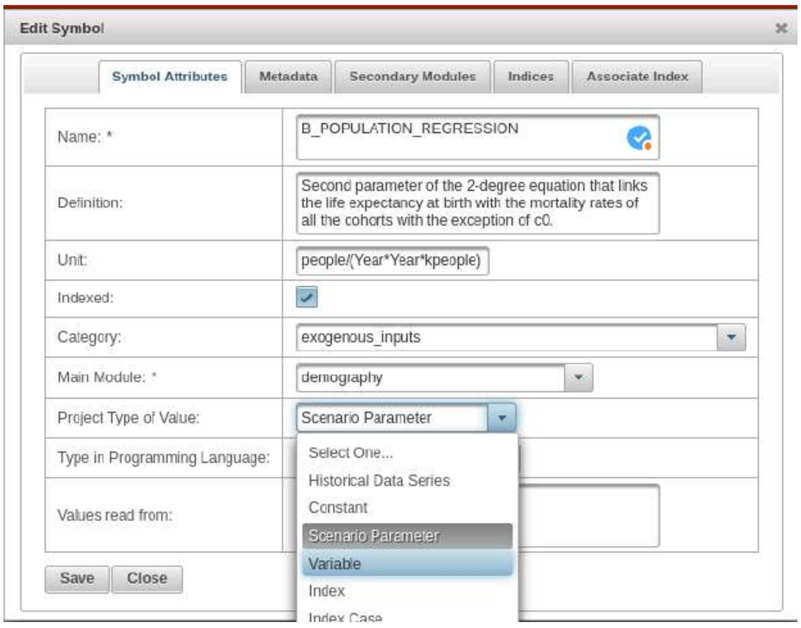


Figure 63: Selecting a Symbol from the Symbols list.

4. In the newly opened page, identify the “Project Type of Value” field, and change it to either *Scenario Parameter* or *Variable*.



Edit Symbol

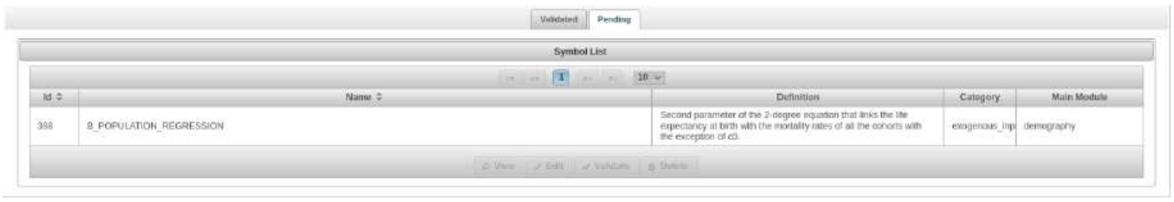
Symbol Attributes | Metadata | Secondary Modules | Indices | Associate Index

Name: *	B_POPULATION_REGRESSION
Definition:	Second parameter of the 2-degree equation that links the life expectancy at birth with the mortality rates of all the cohorts with the exception of c0.
Unit:	people/(Year*Year*kpeople)
Indexed:	✓
Category:	exogenous_inputs
Main Module: *	demography
Project Type of Value:	Scenario Parameter
Type in Programming Language:	
Values read from:	

Figure 64: Project Type of Value.

Symbols that have their Project Type of Value set to *Scenario Parameter*, will be configurable from the Desktop app while Symbols defined as *Variable*, will be available as model outputs to plot.

5. Click on the save button at the bottom of the screen (see Figure 64) and then click on Close.
6. Click again on the Symbol from the list (as in point 3), and click on Edit again. Note that the Symbol may have moved to the "Pending" tab, if it had already been validated (see Figure 65).



Id	Name	Definition	Category	Main Module
388	B_POPULATION_REGRESSION	Second parameter of the 2-degree equation that links the life expectancy at birth with the mortality rates of all the cohorts with the exception of c0.	exogenous_inp	demography

Figure 65: Validated and Pending tabs on the Symbols list.

7. Once inside the Symbol configuration menu, you will see that the *Input/Output Attributes* tab was added to the top menu (Figure 66). This menu page will look differently, for *Project Type of Value Scenario Parameter* or *Variable*. From here on, we will be looking only at the interface for *Scenario Parameter*, which will also cover all fields that are available in the *Variables* interface.

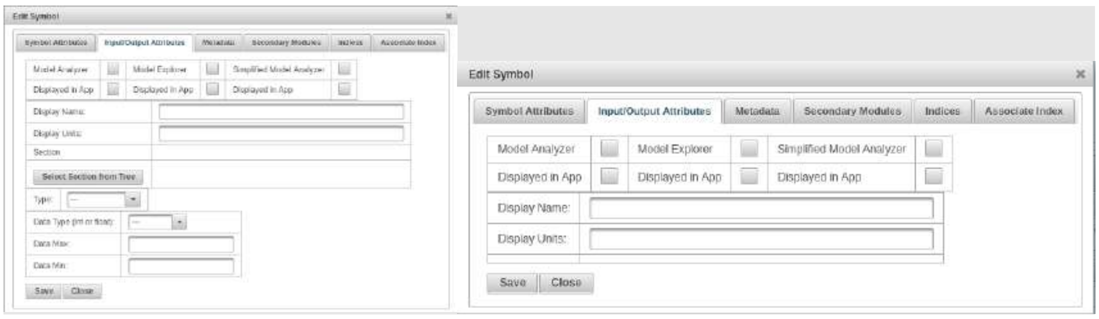
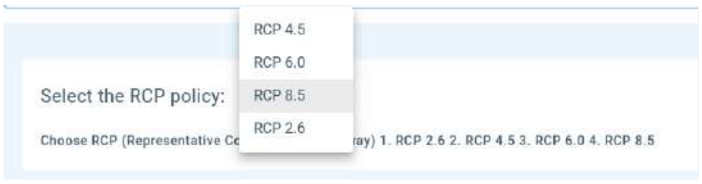


Figure 66: *Input/Output Attributes* tab for Scenario Parameter (left) and Variable (right).

8. The first row of checkboxes in the Input/Output Attributes tab defines whether that Symbol will be included in the app's scenario parameters/output variables.

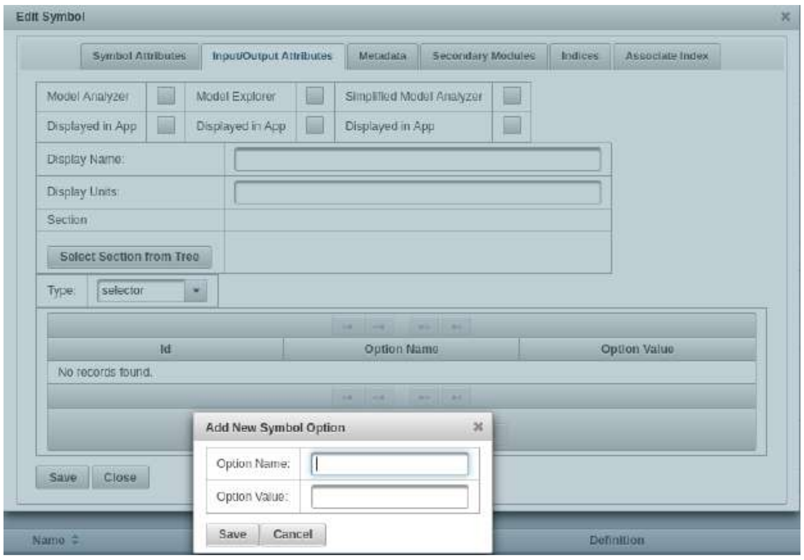
9. The second row of checkboxes controls whether the Symbol will be displayed in the app. In general, if the checkbox corresponding to a certain app is checked on the first row, it should also be checked in the second, unless we want to prevent users from modifying that Symbol for some reason.
10. The next field is Display Name, in which we will write a name for that Symbol which is easier to understand than the parameter name used inside the WILIAM model.
11. The next field is Display Units, which is used for the exact same purpose than the Display Name field.
12. Only in the Variables interface, there is a field named Indicator, which tells the app whether this output variable should be plotted by default or not. This option is used to automatically display relevant simulation outputs when the first simulation results become available.
13. By clicking on the Select Section from Tree, we will be able to group *Scenario Parameters* together in Sections and subsections. In the interface, variables will be organized according to the module they belong to in the WILIAM model, and the Section and Subsections that we select here.
14. In the Type dropdown, we can select between 5 types:
    a. Selector: if we chose this type, we allow users to select different options for that specific *Scenario Parameter*, by means of a dropdown (Figure 67).


Select the RCP policy:

Choose RCP (Representative Co ray) 1. RCP 2.6 2. RCP 4.5 3. RCP 6.0 4. RCP 8.5

Figure 67: Selector widget in the Model Analyzer GUI.

The options and the values associated to it need to be provided in the Input/Output Attributes tab, by clicking on the Create button (Figure 68).


Edit Symbol

Model Analyzer ☐ Model Explorer ☐ Simplified Model Analyzer ☐
Displayed in App ☐ Displayed in App ☐ Displayed in App ☐
Display Name:
Display Units:
Section
Type: selector

Id	Option Name	Option Value
No records found.		

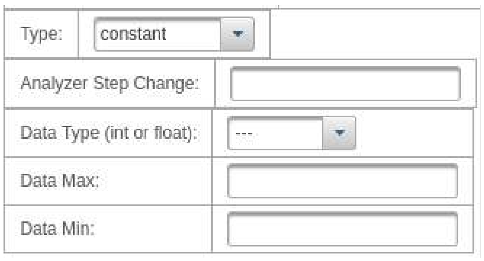
Add New Symbol Option
Option Name:
Option Value:

Figure 68: Data Client menu where options of the selectors are configured.

It is important to note that the values given to *Option value* must match those used in the WILIAM model.

b. Switch: is used for binary parameters (yes/no, activated/deactivated, on/off). In the model, *On* corresponds to 1 and *Off* to 0.

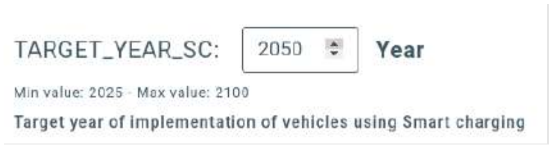
c. Constant: this is used for parameters that are defined with GET_DIRECT_CONSTANTS in the Vensim® model. When selecting this type, you will need to fill in the following fields (Figure 69):



Type:	constant
Analyzer Step Change:	
Data Type (int or float):	---
Data Max:	
Data Min:	

Figure 69: Fields to fill in when selecting *constant* as Type in the Input/Output Attributes tab.

The Analyzer *Step Change* field is used to define the change in the value of the parameter that will occur every time the user presses on the up or down arrows (to change the value of the parameter without having to type it) (Figure 70).



TARGET_YEAR_SC: 2050 Year

Min value: 2025 · Max value: 2100

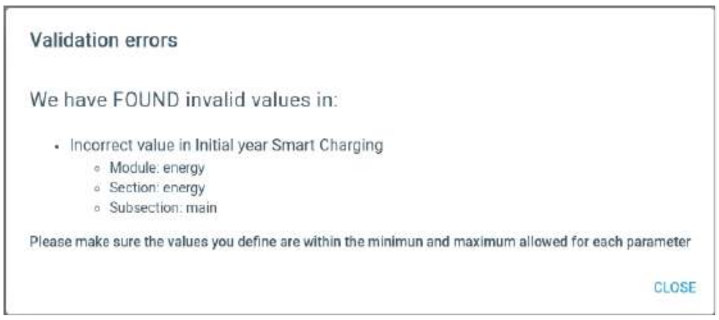
Target year of implementation of vehicles using Smart charging

Figure 70: Changing the value of a parameter using the arrows in the Model Analyzer interface.

By no means this implies that the user is constrained to define values that are multiples of the step defined here. Note that if the value of the step is not provided in the Data Client, the default step will be set to 1, which is not ideal neither for parameters whose values range between –1 and 1, nor for parameters with values spanning over several orders of magnitude (e.g., 1 to 10000).

The *Data Type (int or float)* dropdown in Figure 69 allows to define whether decimals should be displayed (float) or not (int) in the interface for the values of that Symbol (e.g., if the values of that Symbol correspond to years, they should be defined as int, to prevent showing them as 2020.0 in the interface).

The Data Max and Data Min are used to set minimum and maximum values for that Symbol (Figure 69). If the user defined a value below the Data Min or above the Data Max, they will be warned that the value is not correct (Figure 71).



Validation errors

We have FOUND invalid values in:

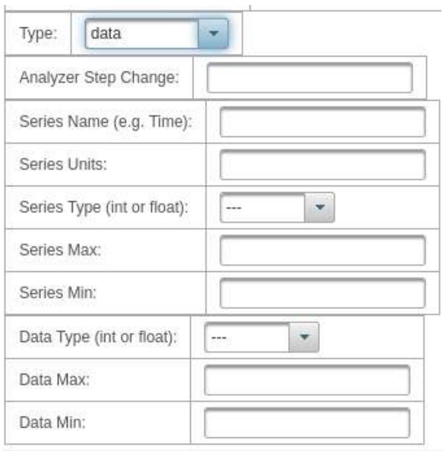
- Incorrect value in Initial year Smart Charging
  - Module: energy
  - Section: energy
  - Subsection: main

Please make sure the values you define are within the minimun and maximum allowed for each parameter

Figure 71: Value outside bounds warning.

Note that the bounds of each parameter should be defined within the range of convergence of the model, to avoid situations where the WILIAM model would fail to converge.

d. Data: this is used for parameters that are defined with GET_DIRECT_DATA in the Vensim® model. Here, we need to fill all fields in Figure 72.



Type:	data
Analyzer Step Change:	
Series Name (e.g. Time):	
Series Units:	
Series Type (int or float):	---
Series Max:	
Series Min:	
Data Type (int or float):	---
Data Max:	
Data Min:	

Figure 72: Fields to fill in for Type data.

Here we only cover those fields that were not covered when the Type is set to constant (point c above).

The Series Name (e.g., Time) field in Figure 72 defines the independent variable (x). When the parameter is defined as GET_DIRECT_DATA, the independent variable is always Time, but for GET_DIRECT_LOOKUP it could be anything else.

The Series Units are the units of the independent variable (usually years).

Series Max and Series Min are like Data Max and Data Min, but applied for the independent variable.

e. Lookup: this is used for parameters that are defined with GET_DIRECT_LOOKUPS in the Vensim® model. The fields to fill in are the same as for data Type.

3.8.2. CODE OF THE MODEL ANALYZER

The code to deploy the Model Analyzer is maintained in two separate repositories:

3.8.3. THE MODEL ANALYZER REPOSITORY

This repository (https://github.com/locomotionH2020/model_analyzer) hosts the open-source backend and Electron desktop app for the Model Analyzer.

The backend handles the following tasks:

1. Manage user access (registration, login, etc.) and user data (simulation results and scenarios).
2. Queues requests from the user's Desktop app.
3. Runs the compiled Vensim® model, passing *cin* and *tab* files as inputs to modify values of *Scenario Parameters* requested by the users.
4. Sends back the simulation results to the Desktop app when the users request them.

The frontend is the interface with the users. The functionality of the Desktop app is presented in Section 0. The executables for the Desktop app are automatically built in the repository using Github actions.

3.8.4. THE SCENARIO BUILDER

This repository (https://github.com/locomotionH2020/model_analyzer_scenario_builder) contains the code to generate the files that will need to be placed in the Windows server for the model to run correctly.

The *generate_jsons.py* script performs the following tasks:

1. Parses user input. The first argument corresponds to the app for which the input files need to be generated (analyzer, analyzer_simplified, explorer) and the second argument corresponds to the database from which the metadata will be retrieved (prod or test).
2. Connects to the Data Client and obtains the scenario configuration (for analyzer, analyzer_simplified or explorer)
3. Gets the value of the latest tag in the WILIAM repository (model version) using the Gitlab API (reads Gitlab credentials from environmental variables).
4. Loads the WILIAM.mdl model using PySD and initializes all external data objects.
5. Checks that the metadata downloaded from the Data Client for each *Scenario Parameter* and output *Variable* (type, units, dimensions, etc.) is consistent with the values in the model file and logs any potential conflicts in the *log.txt* file.
6. Adds default values for the *Scenario Parameters* defined in the Data Client inside the JSON file downloaded from it and writes a new *inputs_(analyzer/analyzer_simplified/explorer).json* file. Note that the Data Client only has metadata, hence the default parameter values must be collected from the initialized model file.
7. Rewrites the mdl file, replacing the data after the equal sign for equations that load external data using GET_DIRECT_DATA. This is needed so that Vensim® can read the values for this parameter from a tab file.
8. Creates one *lst* (format read by Vensim®) file for each output variable configured in the Data Client.

   This repository also includes a Jupyter notebook to verify the consistency of the Data Client with those in the WILIAM model).

3.9. DEPLOYMENT AND OPERATION INSTRUCTIONS

3.9.1. ENVIRONMENTAL VARIABLES

The following environmental variables need to be defined in the Windows server. They are already configured, so it would only be necessary to create them if the deployment is made on a different server.

- *NATS_ENDPOINT*: Nats JStream host
- *DESTINATION_FOLDER*: folder from which the GNU/Linux server reads the simulation results (samba mount point)
- *MODELS_PATH*: path to the folder where the mdl models reside
- *LST_FILES_PATH:* path to the folder where the lst files (one for each output variable) are located
- *INPUT_FILES_PATH*: path where the *cin* and *tab* files read by Vensim® are located
- *RESULTS_SIMULATIONS_PATH*: folder where simulation results are stored (must be shared with the GNU/Linux server)
- *VENSIM_PATH*: path to the Vensim® executable (exe file).

To generate the JSONs using the *generate_jsons.py* script, the following additional environmental variables must be defined in the machine where the code is run:

- *API_USER*: username for the LOCOMOTION Data Client.
- *API_PASSWORD*: password for the LOCOMOTION Data Client.
- *GITLAB_TOKEN*: token to use the Gitlab API to access the tag of the WILIAM repository.

3.9.2. UPDATING THE WILIAM MODEL, THE SCENARIOS (METADATA IN THE DATA DICTIONARY) OR BOTH

To create the ***inputs_(analyzer/analyzer_simplified).json***, ***lst files***, ***WILIAM_(analyzer/analyzer_simplified).mdl*** for the *analyzer* or *analyzer_simplified* scenarios, the *generate_jsons.py* script needs to be run.

The instructions given here, are assuming that the *generate_jsons.py script* is run in the Windows server, where the wp11-analyzer-jsons repository is already cloned:

1. Log in to the Windows server with any user.
2. Open the anaconda prompt.
3. cd to *E:\Development\wp11-analyzer-jsons.*
4. cd to the wiliam folder (*E:\Development\wp11-analyzer-jsons\wiliam*) and pull the latest version of WILIAM using:

   ```
   git pull
   ```

5. cd to the folder where the generate_jsons.py file is located (*E:\Development\wp11-analyzer-jsons*).
6. Activate the virtual environment with*:*

   ```
   conda activate analyzer-jsons
   ```

7. Generate the files for the scenario ***analyzer*** or ***analyzer_simplified*** using the following command:

   ```
   python generate_jsons.py analyzer(analyzer_simplified) prod(test)
   ```

   The first argument is to indicate whether to create the analyzer (full) or the analyzer_simplified scenario inputs.json file. The second argument (prod/test) is used to indicate whether we want to collect the metadata from the production Data Client of from the test Data Client.

8. Follow the instructions on the anaconda prompt. If changes were made in the Data Client, it will ask you to introduce a new version number. This version number will be shown in the Electron application (trying to compare scenarios with different version numbers in the Electron app will fail). Note that it will ask you if you want to use any existing translation of the WILIAM model (if there is a WILIAM.py in the *wiliam* folder) or if you want to translate the model again. **If changes were made to WILIAM.mdl or any of the Excel files, you should opt to run the translation.**
9. When the script finishes, cd to the results folder (*E:\Development\wp11-analyzer-jsons\results*). Inside this folder, you should see a folder, named after each scenario (analyzer and analyzer_simplified).
10. cd into the folder corresponding to the scenario you passed as first argument to the *generate_jsons.py* script. Inside, you should see several files, including a *log.txt* file. Inspect this file to see any potential coherence issues between the metadata in the Data Client and the model equations. If there are any errors, fix them, and run the script again. Else, move to the next step.
11. The files inside this folder (except for *log.txt*) need to be moved to the Windows server (replacing the preexisting ones), in the following paths:
    a. *inputs_analyzer.json -> E:\Models\inputs*
    b. *WILIAM_analyzer.mdl -> E:\VensimData\models*
    c. *All lst files inside the lst_files folder -> E:\VensimData\models\lst_files*
12. Although the *generate_jsons.py* script does not change the *parameters* nor *scenario_parameters* folders from the WILIAM model, if the files inside have been modified, they have to be copied to *E:\VensimData\models* (replacing the existing folders with the same names).

13. When the *mdl* model file is updated, the *dll* file located in *E:\VensimData\models (e.g., WILIAM_analyzer_simdp_x64.dll)*, needs to be deleted. If this is not done, an *Unknown software exception* will be raised by Vensim® when running a simulation.

3.9.3. BACKEND DEPLOYMENT (LINUX SERVER)

The Docker images are built from the Dockerfile that exist in the repository. You need to build the Dockerfile-base, and then the Dockerfile. The Dockerfile-base image is used by the Dockerfile, to save time when installing libraries using PIP.

1. Login to the Linux server with the superadmin user.

ssh superadmin@linux_server_ip

2. cd to/Development/iskra/model_analyzer
3. Run git pull, to get the latest updates in the code
4. Change the docker images inside of the docker_compose.yaml
5. Run docker compose:

   docker-compose up -d

6. The previous command will start Postgress, redis, NATS and Guillotina.

Once the Docker images are deployed in the Linux server, login to the Windows server to run the workers.

Note that it is not necessary to execute the workers script every time the Guillotina image (backend) is deployed (using *docker-compose up –d*), since the two of them work independently.

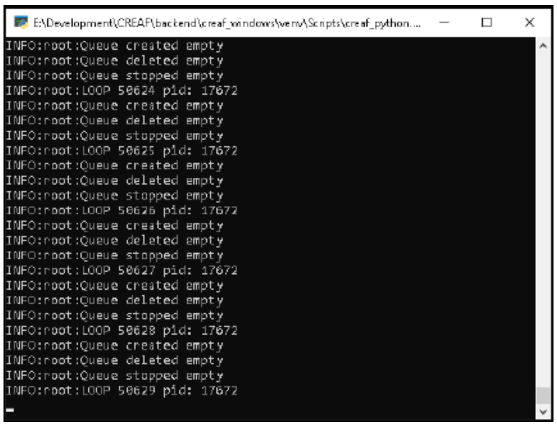
3.9.4. RUNNING WORKERS (WINDOWS SERVER)

Workers control how many Vensim® instances can be run in parallel in the Windows server. Here are the instructions on how to run any number of them in the Windows server:

1. Open an Anaconda powershell prompt
2. cd to the directory where the virtualenv of the CREAF project is (E:\Development\CREAF\backend\creaf_windows\creaf_windows)
3. Execute the *run_workers.py* python script:

   \venv\Scripts\creaf_pythonw.exe .\run_workers.py N

   where *N* is the number of workers that will start the Vensim® processes. When the previous command is run, N terminal windows will open, which will display all their activity (Figure 73).



```
E:\Development\CREAF\backend\creaf_windows\venv\Scripts\creaf_python....
INFO:root:Queue created empty
INFO:root:Queue deleted empty
INFO:root:Queue stopped empty
INFO:root:LOOP 50624 pid: 17672
INFO:root:Queue created empty
INFO:root:Queue deleted empty
INFO:root:Queue stopped empty
INFO:root:LOOP 50625 pid: 17672
INFO:root:Queue created empty
INFO:root:Queue deleted empty
INFO:root:Queue stopped empty
INFO:root:LOOP 50626 pid: 17672
INFO:root:Queue created empty
INFO:root:Queue deleted empty
INFO:root:Queue stopped empty
INFO:root:LOOP 50627 pid: 17672
INFO:root:Queue created empty
INFO:root:Queue deleted empty
INFO:root:Queue stopped empty
INFO:root:LOOP 50628 pid: 17672
INFO:root:Queue created empty
INFO:root:Queue deleted empty
INFO:root:Queue stopped empty
INFO:root:LOOP 50629 pid: 17672
_
```

Figure 73: Worker in action.

If the number of workers (N) passed as the argument to the script, is different than the current number of running workers, it will stop/start the number of workers corresponding to the difference between the two values. Hence, note that if N workers are running, and this script is run with X < N, Y process will be deleted, where Y = N – X.

Note also that the terminal windows of the workers should always be left open, as closing them would kill the workers.

To know how many processes are running, run the same script without passing any arguments:

```
\venv\Scripts\creaf_pythonw.exe .\run_workers.py
```

3.9.5. BUILDING THE ELECTRON APP INSTALLERS AUTOMATICALLY WITH GITHUB ACTIONS

New versions of the Model Analyzer installers for the different platforms (GNU/Linux, Windows and Mac) are automatically built every time a new tag is created in the project repository.

For tags to create new releases, the version number must be preceded with a “v”. For example, the following command would generate a tag for version 1.4.0:

```
git tag -a v1.4.0 -m "my version 1.4"
```

The executables can be downloaded from the releases section in the Github repository (https://github.com/locomotionH2020/model_analyzer/releases).

3.9.6. BUILDING THE ELECTRON APP INSTALLERS (FOR YOUR CURRENT OS) MANUALLY

With this approach, you can create the executables directly in your machine. Note that you can only create executables for your current OS (if you are on Windows, you can only create .exe files). Note as well that to create the executables **you must have *npm* package manager for Node.js installed on your machine**.

From the project folder:

1. cd to the *desktop* folder
2. Run the following commands:
   a. *npm i*
   b. *npm run make*

This will put the new executables in a folder named after your operative system executable file format **inside the desktop/out/make folder**.

The app can also be run in “development mode” (no executables created) for debugging purposes. To do so, from the *desktop* folder, run:

a. *npm i*
b. *npm run dev*

3.9.7. MANAGING USERS ACCOUNTS FROM GUILLOTINA INTERFACE

Guillotina is an API Rest made with AsyncIO (https://guillotina.readthedocs.io/en/latest/).

To authenticate the calls:

- Basic auth. Username: root
- password: {/secrets/root_password.txt}

Where root password is in the Linux machine. This will assign root privileges to the user.

3.9.7.1. LISTING ALL USERS

GET http://proxy.geeds.es/data/container/users/@items

To paginate the results:

GET http://proxy.geeds.es/data/container/users/@items&page_size=20&page=2

3.9.7.2. CREATING USER USING EMAIL VALIDATION

POST http://proxy.geeds.es/data/container/@users

```
{
        "@type": "User",
        "id": "foo_user@iskra.cat",
        "username": "foo_user@iskra.cat",
        "name": "Foo User",
        "email": "foo_user@iskra.cat",
        "password": "foo_user1234",
        "user_roles": ["guillotina.Member"]
    }
```

3.9.7.3. CREATING A USER WITHOUT EMAIL VALIDATION

POST http://proxy.geeds.es/data/container/users

```
{
        "@type": "User",
        "id": "foo_user@iskra.cat",
        "username": "foo_user@iskra.cat",
        "name": "Foo User",
        "email": "foo_user@iskra.cat",
        "password": "foo_user1234",
        "user_roles": ["guillotina.Member"]
    }
```

3.9.7.4. DELETING A USER

Once you have listed all users, you can delete one, the id comes from the @id field of List Users

DELETE http://proxy.geeds.es/data/container/users/{foo_user}

3.9.7.5. GETTING A USER

GET http://proxy.geeds.es/data/container/users/{foo_user}

3.9.7.6. CHANGING USERS DISK QUOTA

To change the quota disk for all the new users that are going to be created, there are two parameters in the config-dev.yaml named max_size_allowed and max_in_queue_scenarios, one restricts the maximum size a user can use, and the other the maximum number of simulations that can queue. By default, these are set to 5Gb and 5, respectively. After changing these values, the docker image needs to be built and deployed.

In order to change the quota of an already created user, a PATCH can be done:

PATCH: http://proxy.geeds.es/data/container/users/n.bacardit@iskra.cat

```
{
  "creaf.content.behaviors.IScenariosBehavior": {
    "max_size_allowed": "1524288000"
}}
```

Now the user n.bacardit@iskra.cat has about 15GB to use. To check the user quotas a GET can be done on the user:

GET: http://proxy.geeds.es/data/container/users/n.bacardit@iskra.cat

Among other properties, the user will find:

```
creaf.content.behaviors.IScenariosBehavior": {
"size_used": 11673755,
"max_size_allowed": 1524288000,
"max_in_queue_scenarios": 5
},
```

The user has used 11Mb, has 15GB of quota and can queue up to 5 simulations.

3.9.8. DEBUGGING THE LINUX SERVER

If any anomaly is detected in the Linux server, the best place to start looking are the logs of the Docker containers. To do so, ssh into the Linux machine and run the following commands:

1. docker ps (to show running containers by default)
2. docker logs (to retrieve logs present at the time of execution)

Additionally, to debug the code in the Docker image, the following command may be issued:

docker exec –it {id_container} bash

4. CROSSROADS II GAME

4.1. IDENTIFIED REQUIREMENTS

Crossroads II is an electronic version of the game Crossroads[1] [5], a cooperative and social game that is made in groups, that takes place in person by filling out paper forms. The form completed by each team is given to the moderator who leads the game proposals in a simulator. The moderator enters the answers provided by the players into a simulator and returns graphs as an answer. The forms are proposals for policy measures to attack the problem of climate change and the graphs show the evolution of the economy and the climate. These simulations take several minutes, since each time the data is entered, a simulation is performed. If there are several groups playing at the same time, this means a fairly high waiting time. The main idea of Crossroads II was to streamline interaction while allowing the development of online games.

Two types of roles are distinguished in the game, the role of participating players and the moderator role. Participating players always play in groups. A game takes place in a room with a moderator and players

[1] Capellán-Pérez, I., Álvarez-Antelo, D., Miguel, L.J. (2019). Global sustainability Crossroads: A participatory simulation game to educate in the energy and sustainability challenges of the 21st century. Sustainability, 11, 3672. https://doi.org/10.3390/su11133672

who are distributed in groups. The distribution in groups can be done by selection or automatically. The moderator must create the room where the game will take place. The game will be divided into rounds.

Three phases are distinguished in the game: the explanation phase, the intermediate phase and the final phase. In the explanation phase and first round, the moderator explains to the participants how the game works; groups discuss answers and answer questions in the app itself; The moderator will also have access to the application by controlling the game; This phase is performed only once. In the intermediate phase, the groups receive the results of the simulation; the results obtained are discussed among all participants; and groups answer the questions again for the next iteration; This phase is repeated as many times as rounds are made. In the final phase, the groups receive the results of the last simulation; the results are discussed among all; Commonly, the moderator shows a scenario with positive results; This phase is only done once at the end of each game.

The main difference between the original game and the electronic format is that it is no longer necessary to use the moderator as an intermediary, speeding up the game, and allowing the different players to make their proposals without depending on the moderator to see the results. Other notable differences are: a) that the argumentation in the group is made in a chat between the players of the group, b) the members of the groups must reach a consensus through the application, and c) game elements are introduced to positively reinforce the work of the group and to compete with the rest of the groups playing in the room, for example rankings and awards.

The main sources for the elicitation of requirements were the description of the initial version of Crossroads, interviews with the authors of the game and with researchers of the LOCOMOTION project, as well as with the opinions of the group of professors of the ECO-profes interest group.

4.2. MOCKUPS

The app requires the game moderator to be registered. This provides a simple interface that is shown the following Figure 74.

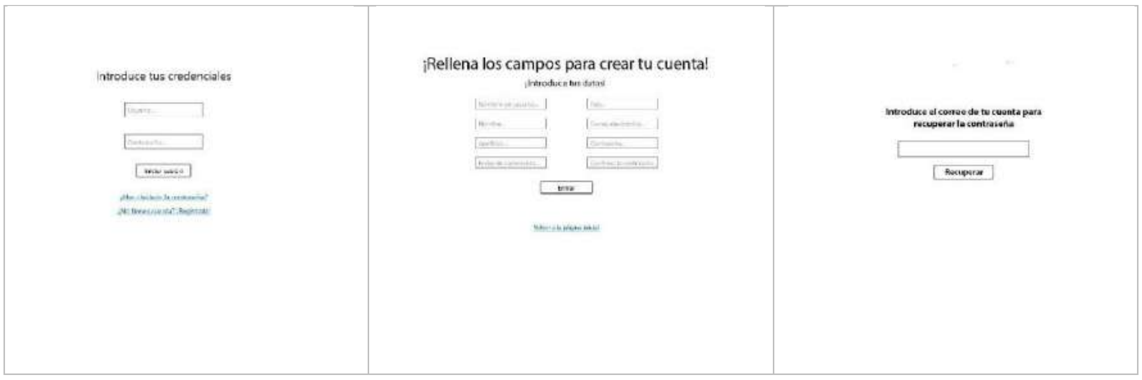

Introduce tus credenciales
¡Rellena los campos para crear tu cuenta!
¡Introduce tus datos!
Introduce el correo de tu cuenta para recuperar la contraseña
Recuperar

Figure 74: Welcome page for a player and a moderator.

Once registered, the moderator can organize games, for which he creates game rooms by setting the number of groups and the maximum number of players per group. We show below (Figure 75) the mockups for such interaction.

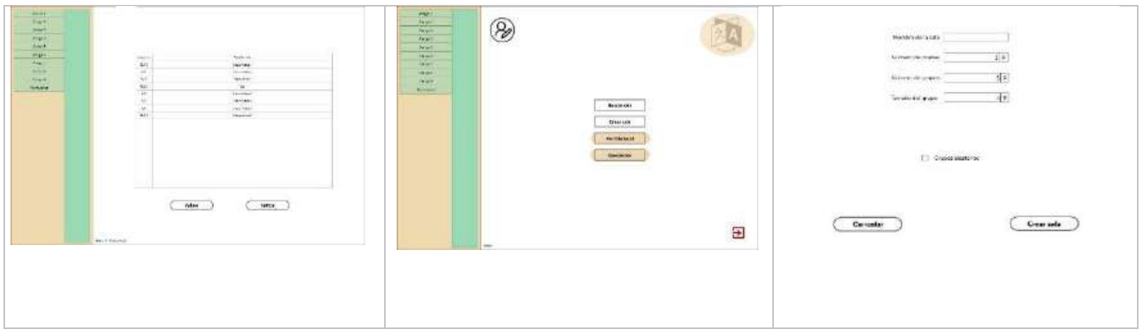


Figure 75: Groups definition and room creation.

The moderator, once the room has been created, generates a game code, which is a key with which the players can participate in the game. He also has a control panel to monitor the evolution of the game (Figure 76).

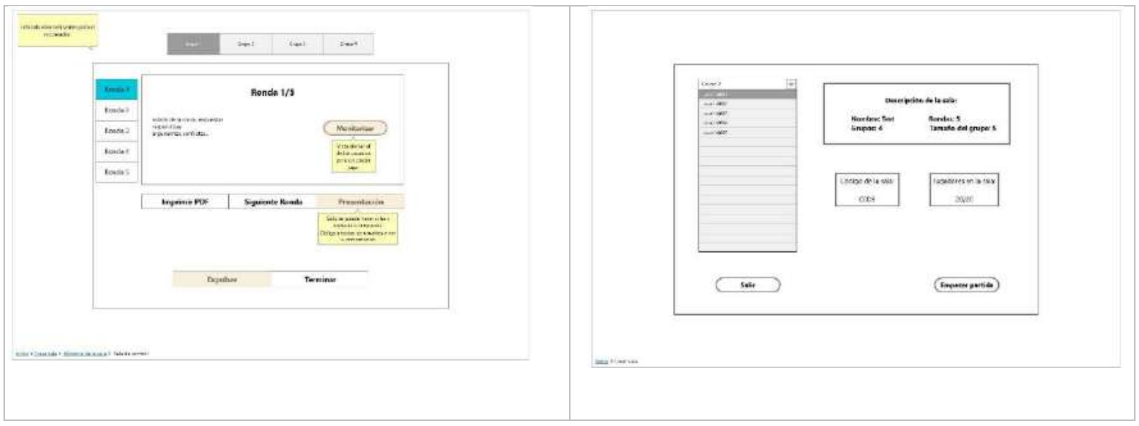


Figure 76: Control views to monitor the evolution of the game.

The player enters the game with a game key. Once inside, he must choose a group with which he will make his proposals. The groups compete against each other. Figure 77 represent the interface for choosing a group.

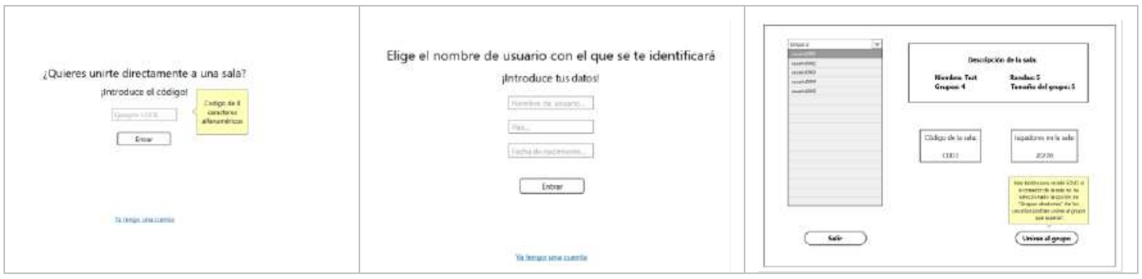
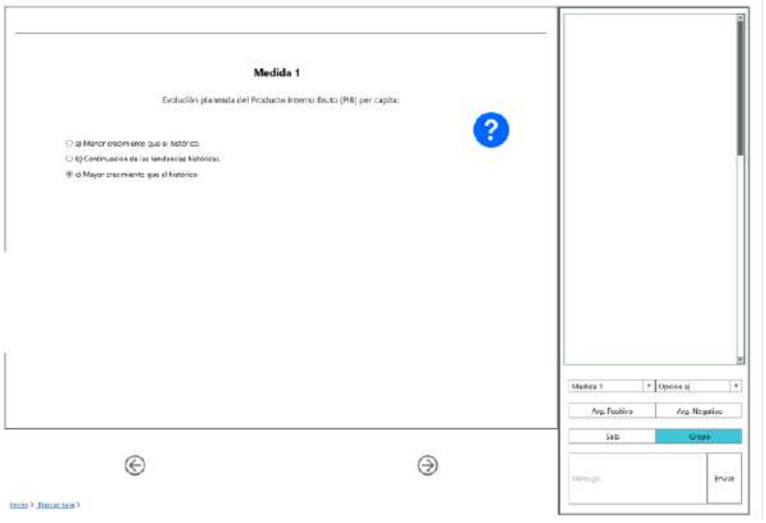


Figure 77: Interface for choosing a group entering your user name.

In the previous phase, the players have access to a help that sets the game. When the moderator starts the game, the players can complete the formulary, which are the political measures they propose to solve the problem of climate change. In the formulary, players have a chat where they can share their opinions with the other participants in the game (Figure 78).



Medida 1

Evolución planeada del Producto Interno Bruto (PIB) per capita:

- ( ) a) Menor crecimiento que el histórico.
- ( ) b) Continuación de las tendencias históricas.
- [✓ c) Mayor crecimiento que el histórico.]

Figure 78: Main view to select a policy value and chat with the group members.

Once all the questions on the form have been completed, the players see a summary of the set of proposals from the group members. They then move on to a phase in which they must resolve the conflicts (Figure 79).

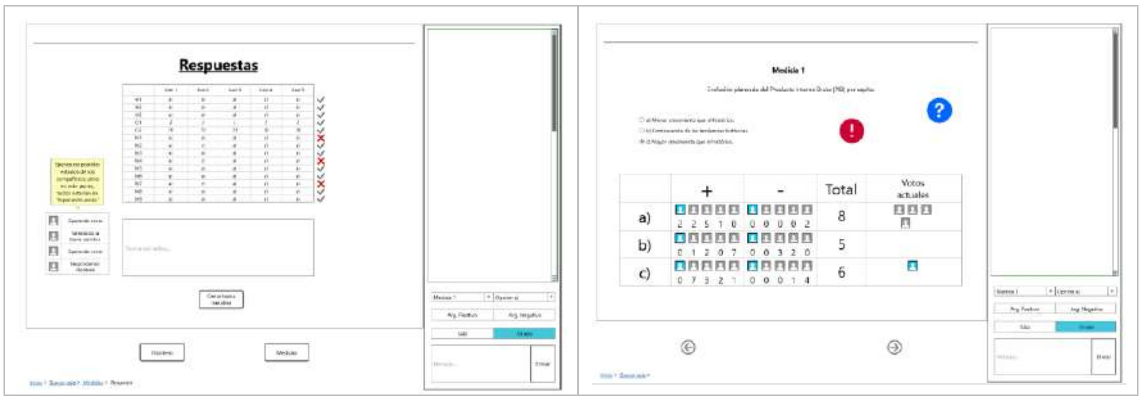


Figure 79: Summary of policy results in the group members and view to solve conflicts.

Once the conflicts have been resolved, players see the effects of their proposals in descriptive graphs. A series of recommendations are suggested to them and they move on to the next round of play. At the end of the rounds, the game shows them how good their results were compared to those obtained by the other teams (Figure 80).

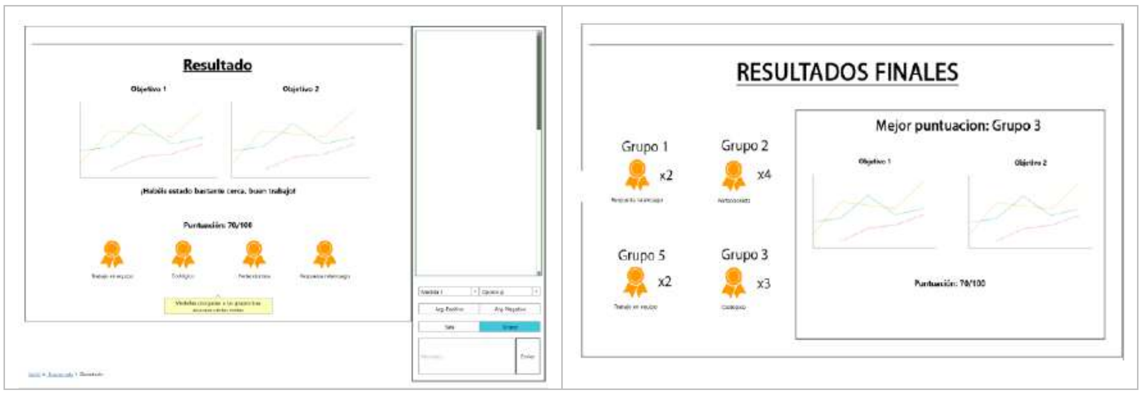
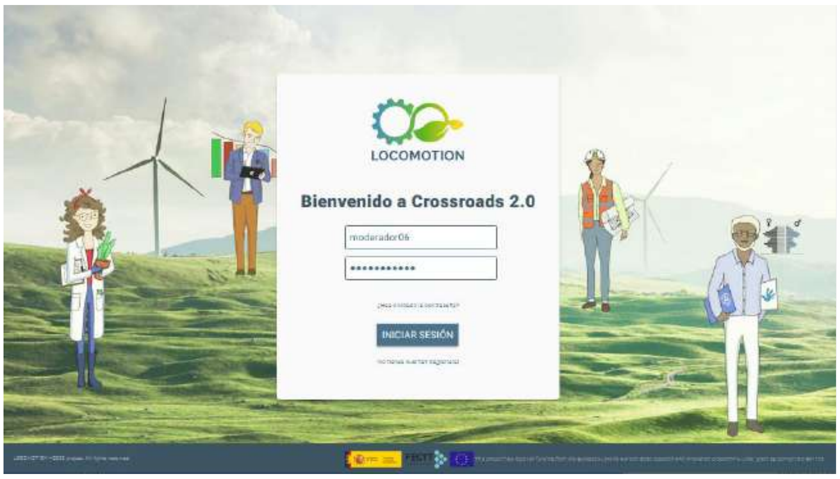


Figure 80: Game results and comparison between groups.

4.3. GUI DESIGN

In this section, the game design of LOCOMOTION is presented. The initial screen (Figure 81) invites you to participate in a game. In order to do so, you must organize a game. To organize a game, you must register as a moderator. If we already have an account to create games, we log in.



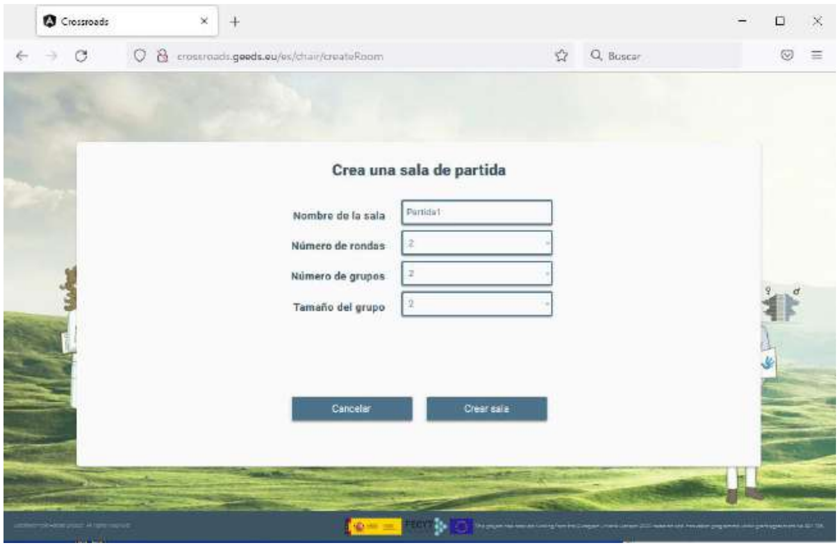
Bienvenido a Crossroads 2.0

moderador06

•••••••••••

Figure 81: Initial screen of Crossroads II Game.

Once you are logged in as a moderator, you are able to create a game by setting the number of rounds, the number of groups and the size of the groups (Figure 82).



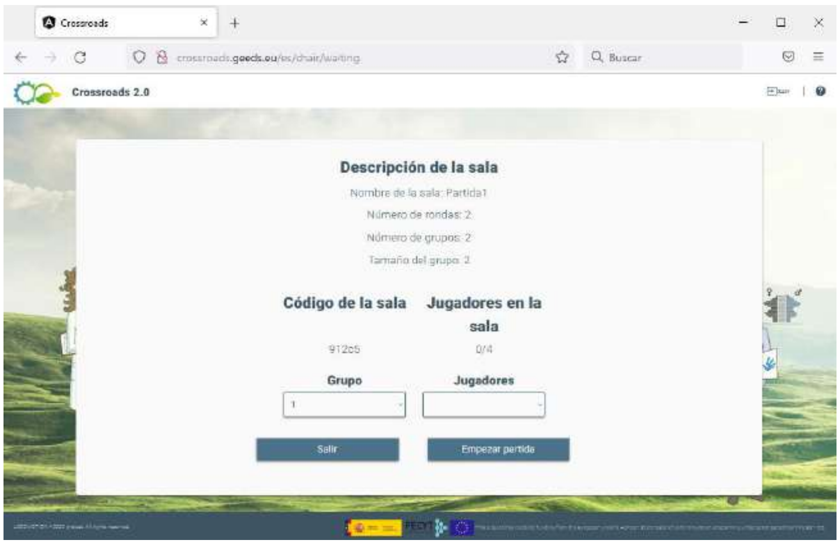
Crea una sala de partida

Nombre de la sala	Partida1
Número de rondas	2
Número de grupos	2
Tamaño del grupo	2

Figure 82: Moderator view to create the game with the room, rounds and groups.

When the game is created, the moderator obtains a room code that players can use to start the game (Figure 83). If you want to play in individual mode, you can open a tab and enter the code after clicking on the PLAY button. You can use the room code to start the game by including it in the app (Figure 84).

Each player enters an identifier and some indicators (Figure 84) and goes to an area where he/she watches a presentation video and chooses the group. You can also choose a game avatar (Figure 85) and wait for the moderator to start the game.



Descripción de la sala

Nombre de la sala: Partida1

Número de rondas: 2

Número de grupos: 2

Tamaño del grupo: 2

Código de la sala	Jugadores en la sala
912c5	0/4

Grupo: 1

Jugadores:

Figure 83: Moderator view to create the game with the room, rounds and groups and the room code.

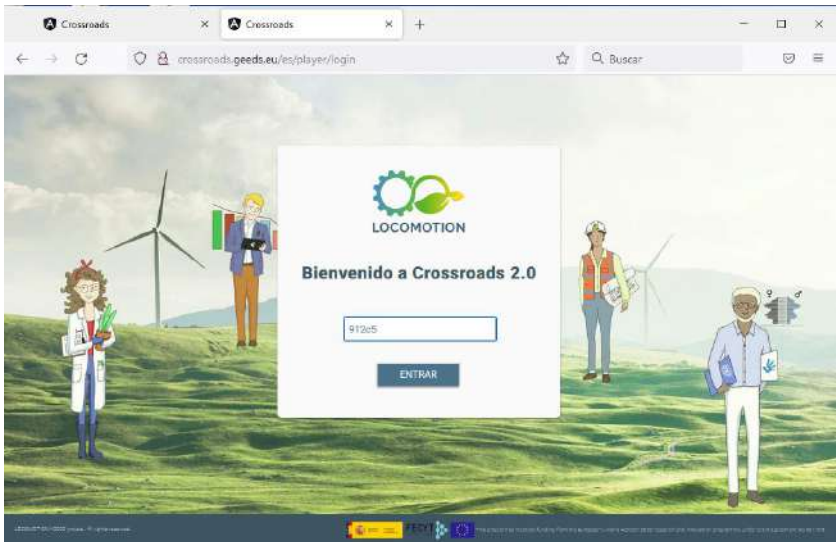

Crossroads
Crossroads
crossroads.geeds.eu/es/player/login
Buscar
LOCOMOTION
Bienvenido a Crossroads 2.0
912c5
ENTRAR

Figure 84: Room code insertion view.

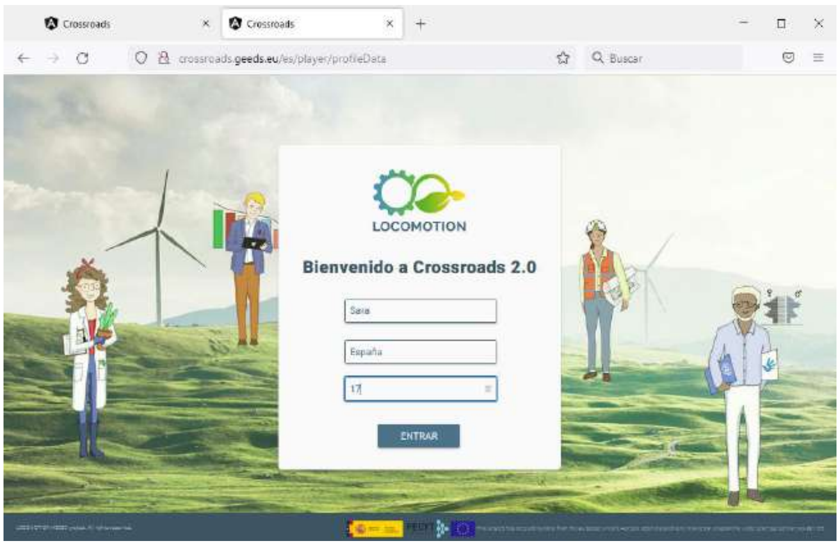
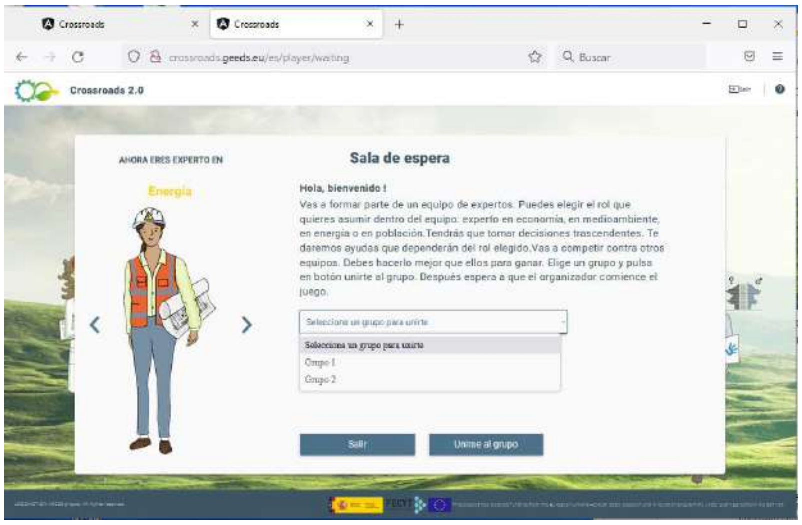

Crossroads
Crossroads
crossroads.geeds.eu/es/player/profileData
Buscar
LOCOMOTION
Bienvenido a Crossroads 2.0
Sara
España
17
ENTRAR

Figure 85: Profile data view.



Sala de espera

Hola, bienvenido !
Vas a formar parte de un equipo de expertos. Puedes elegir el rol que quieres asumir dentro del equipo: experto en economía, en medioambiente, en energía o en población. Tendrás que tomar decisiones trascendentes. Te daremos ayudas que dependerán del rol elegido. Vas a competir contra otros equipos. Debes hacerlo mejor que ellos para ganar. Elige un grupo y pulsa en botón unirte al grupo. Después espera a que el organizador comience el juego.

Selecciona un grupo para unirte
Grupo 1
Grupo 2

Figure 86: Group and avatar selection.

When the moderator starts the game, he/she enters a control room where he/she can see the evolution of the game (Figure 87).

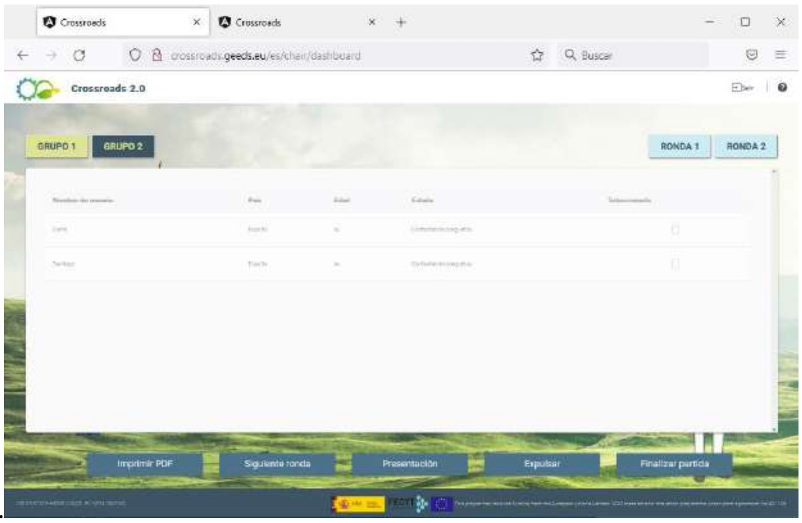


Figure 87: Moderator control dashboard.

The players take a position on the measures to be taken by exchanging opinions in an interactive chat. At the end of the questions, a matrix is displayed in which the conflicts between group members appear (Figure 88).

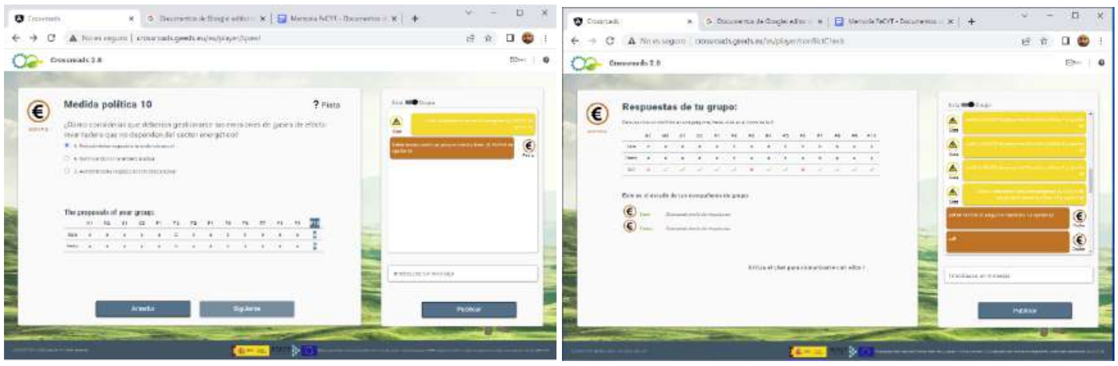


Figure 88: Dashboard to select different policies creating consensus using the interactive chat.

Pressing the red X that you can see in the previous image, a discussion among the group members to resolve the conflicts is opened (Figure 89, left). When the conflicts are resolved, a screen with the results is displayed (Figure 89, right).

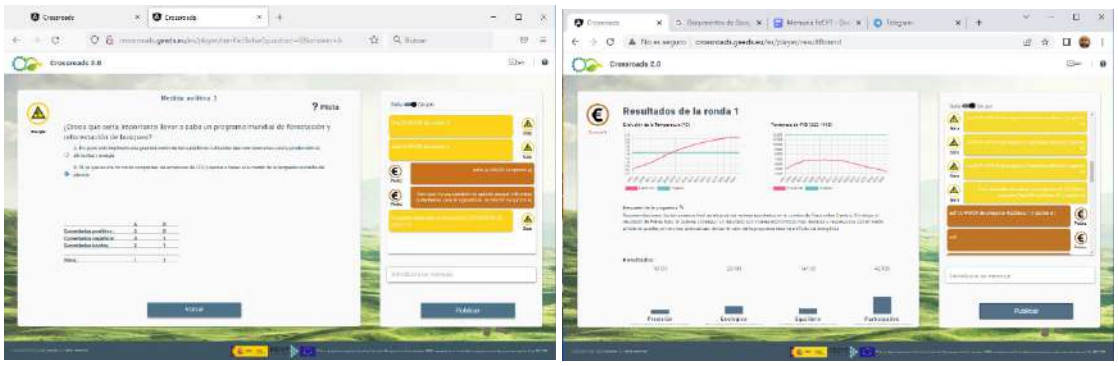


Figure 89: Dashboard to find consensus by interacting again on the selected policy (left) and displayed results after consensus in the selected policies (right).

The moderator can let play more rounds or end the game in the control panel. When the moderator ends the game, the performance of the groups is compared (Figure 90).

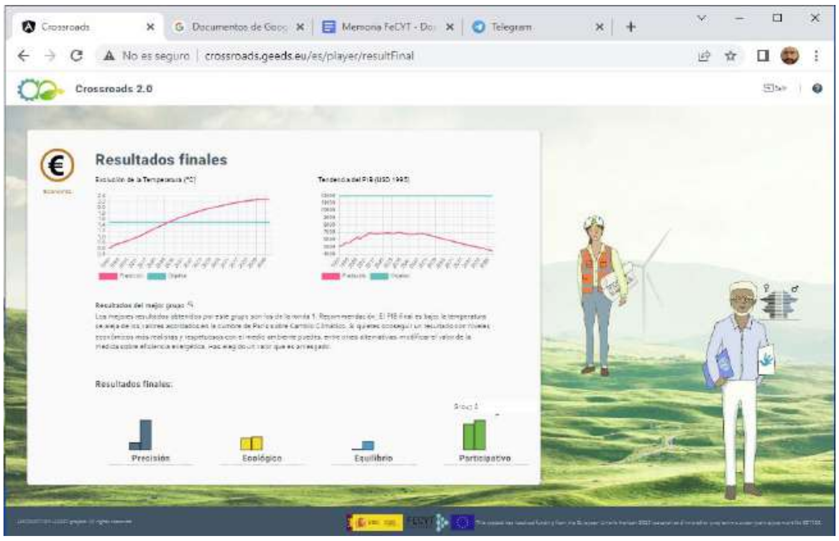


Figure 90: Final view with the game results.

4.4. SELECTED TECNOLOGIES

The backend is implemented with the Springboot framework to define an API for communication with the frontend. The game information is stored in a MySQL database and the result graphs in a MongoDB database. In these cases, MySQL Workbench and MongoDB Compas have been used respectively. Postman was used for backend testing. The development IDE was IntelliJ IDEA.

The frontend is implemented with Angular technology, combining the HTML and CSS views with the logic and communication with the backend using TypeScript. Apart from Node.js and NPM, basic to make a monolithic application in the frontend, the libraries chart.js for the graphics presentation, filesaver.js for the local storage, PrimeNG for the visual components, and Visual Studio were used for the development IDE.

Other secondary technologies are Apache Tomcat as deployment server, Flask and Docker for the enhancement recommendation module, Cypress for unit testing, Git for version and documentation management, the Mockup plus tool, and InVisio for web design definition.

4.5. ARCHITECTURE

The system has been implemented following a client-server architecture as shown in Figure 91.

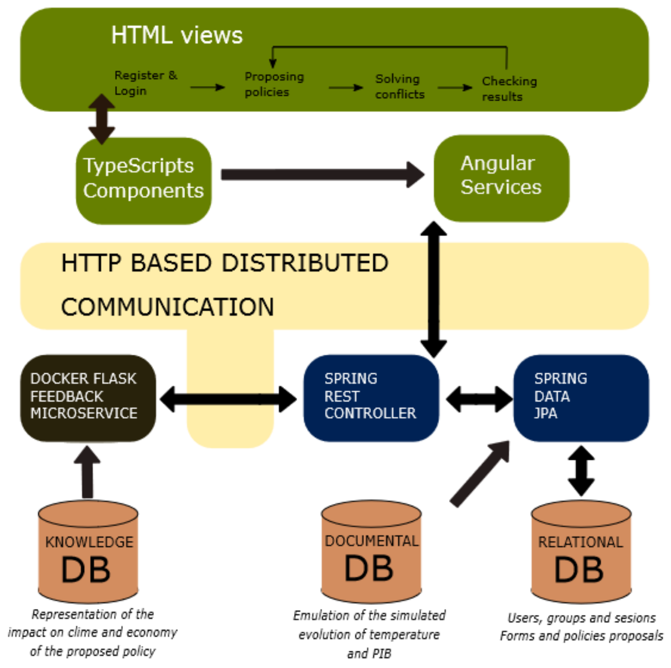

HTML views
Register & Login
Proposing policies
Solving conflicts
Checking results
TypeScripts Components
Angular Services
HTTP BASED DISTRIBUTED COMMUNICATION
DOCKER FLASK FEEDBACK MICROSERVICE
SPRING REST CONTROLLER
SPRING DATA JPA
KNOWLEDGE DB
DOCUMENTAL DB
RELATIONAL DB
Representation of the impact on clime and economy of the proposed policy
Emulation of the simulated evolution of temperature and PIB
Users, groups and sesions Forms and policies proposals

Figure 91: Crossroads II Game architecture.

Users access the application using a web browser. The server with the game data is a server and communicates with the clients using HTTP REST communication services. The communication between the database and the chat, dashboard and discussion tables is implemented with the publish/subscribe paradigm to relieve the server load.

4.6. TOOL IMPLEMENTATION

For the implementation of the tool, the University of Valladolid hired Lucas Matías González Calderón in 2020 for the development of the backend and the first versions of the frontend. Elena Rodriguez Pastor performed the first HTML and CSS implementations of the frontend as part of an internship in 2021. María Galíndo Gómez was hired in 2021 to implement the frontend in Angular, she also made a testing plan. Manuel Alda Peñafiel was hired in 2022 to perform frontend and backend improvements. Adrian Manzano Santos did an internship in 2021 in which he defined the docker with the Recommender System. In 2021, the company Ondeuev designed the visual interface that was translated to HTML and CSS by David Escudero Mancebo. Yania Crespo Carvajal and David Escudero supervised the activities of the Development team.

4.7. QUALITY ASSURANCE

For quality assurance, different types of tests are addressed according to some of the quality sub-characteristics of the ISO 25000 quality model: functional suitability, usability, performance efficiency, and reliabilty.

Regarding the Functional suitability and Reliabilty sub-characteristics, Crossroads functional testing has been divided into two parts, one in charge of testing the back-end and the other in charge of testing the front-end. To check the correct functionality of the back-end, we first performed a battery of black box tests of each of the REST API operations, through Postman, a battery of tests that is launched automatically.

Subsequently, it was decided to automate end-to-end (e2e) tests as well, since these tests allow us to test the application in a similar way as the user would do it. In addition, e2e testing serves to reinforce the results of the back-end testing, since in this type of testing, the user can test the application in a similar way to the user.

back-end testing, since in this type of testing the server is used with its real implementation, instead of using a ``dummy server'' with its expected behavior, as is the case with unit tests.

At the time of project completion, 183 automatic e2e test cases are available, grouped in 18 JavaScript code files and the success rate of these is 100%, i.e., all tests obtain their expected result. As the tests are automatic (due to the nature of e2e testing), each test case will not be detailed here, as the documentation of the test cases resides in the code that performs the tests.

To improve the maintainability of the tests, it was decided to apply the PageObject design pattern, whereby classes are created to encapsulate the access to the interface elements. In this way, the tests are programmed making use of the interface provided by the PageObjects. If the interface changes, only the corresponding PageObject would have to be changed, instead of having to change the implementation of the test cases. For the purpose of making test cases independent of each other and making their results deterministic, a plugin was created for Cypress to delete data from the MySQL database between test case executions. A plugin was also created to delete the contents of a folder, as this behavior was necessary for some test cases that involved downloading a pdf file.

This automated e2e testing process was very positive and achieved sentence coverage measures of over 95% and 80% in branch coverage, a high percentage of coverage for e2e testing in a medium sized application, which is a good indicator of the quality of the testing performed.

In what concerns to usability, the following usability tests (Figure 92 and Figure 93) have been performed:

- Internal test with GEEDS researchers July 15, 2021 in distributed mode, with collection of evaluation questionnaires and proposals for improvement.
- Tordesillas Institute test December 21, 2021 in face-to-face mode, with collection of pre and post questionnaires.
- FeCYT group test February 3, 2022 in online mode, with collection of usability questionnaires.
- Test on March 17, 2022 at the Faculty of Economics EHU in face-to-face mode, with collection of evaluation questionnaires. 20 users aged between 15 and 52 years.
- Test in Locomotion meeting on March 29, 2022 at the University of Valladolid, face-to-face with Project researchers aged between 21 and 58.
- Test at the Pinar Institute on May 27, 2022 in face-to-face mode with exposure by the professor at a single workstation. 23 students between 16 and 18 years old.
- Test EHU Summer Courses July 22, 2022 in face-to-face mode with online monitoring. 17 users enrolled in the Summer Courses between 15 and 66 years old.

Figure 92: Test in Instituto Pinar de la Rubia (left) and test in Instituto de Tordesillas (rigth).

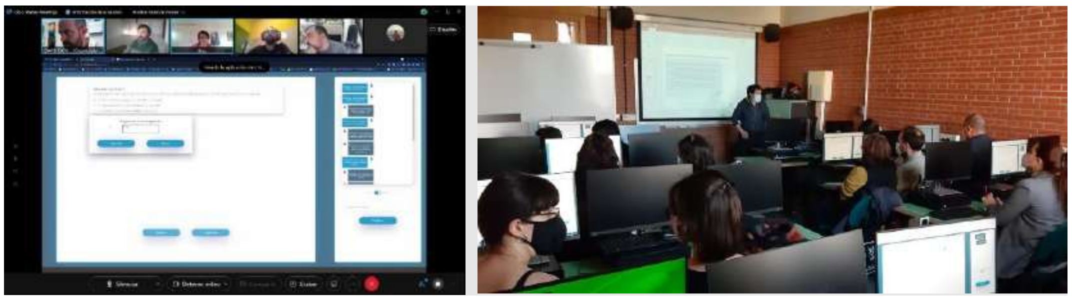


Figure 93: Test with a group of teachers from FeCYT (left) and test in the University of Vasque Country (rigth).

The performance efficiency, and also reliability subfeatures were addressed by designing and executing a load and scalability test suite using the JMeter tool.

Automatic testing of a scenario was prepared in which players (automatically simulated) access a room for a single round game with the code of a previously created room, select a group to join and randomly select a role. Once all players have completed this task, the moderator (automatically simulated) will start the game and all players will answer the questions on the form, so that no conflicts between the answers of all players in each group will be generated. For each question, each player will make a query from one of the associated clues. Once the form is completed by all players in the group, they will display the round results and, upon completion of the game by the moderator, they will display the round results.

Load and scalability tests were run for 30, 50 and 100 players playing a game simultaneously. With the objective of obtaining good performance results that facilitate the playability of 100 people in the same game playing simultaneously, a scenario that could correspond to a group of students of a university degree course.

As performance metrics were collected using JMeter:

- Response time. Speed metric defined as the time interval, in milliseconds, between the end of sending a request and the end of the corresponding system response.
- Productivity. Speed metric that is calculated as the number of requests served per second.
- Utilization. A metric that is defined as the percentage of time a server component is busy relative to the system observation period. CPU and RAM utilization will be calculated.
- Error rate. Average number of requests answered incorrectly and the average number of requests not serviced. The errors produced are classified according to their type and the proportion of each type of error in the total is determined.

Three runs are performed to increase the reliability of the metrics, resulting in a total of 18 runs of the test plan. These 18 runs are performed on three different days, i.e. 6 runs per day, all of them with a fixed number of users. Of these 6 runs, 3 of them were dedicated to closed network tests and the other 3 to open network tests. The tests have always been done at 3 different times of the day for each type of network, in order to increase the independence of the time of day of the data collected.

Other important quality sub-characteristics according to ISO 25000, such as those related to Maintainability, Compatibility and Portability, were approached from the design and implementation by applying best practices in software development, principles and design patterns.

Finally, the Security subfeature was addressed with a Privacy by Design approach whereby only one type of user should register: moderators, as this role creates games and invites users to participate. Hardly any personal data is stored other than what is strictly necessary to retrieve the password via e-mail. For the identification and protection of transactions and data, the technology provided by Spring boot is used. In the case of players, data is only collected for statistical purposes, but it is not possible to retrieve from the stored information who the participant is. That data would be known personally by the moderator who organizes the game: for example, your students, and it is his responsibility to know the correspondence nickname - person who participates. The system would not know more about the participants than the nickname, country and age of the players.

4.8. DEVELOPER GUIDE AND DEPLOYMENT

4.8.1. SYSTEMS

This section shows the different machines that are used to perform the development and/or deployment of Crossroads 2.0. For each of the machines, a description, the connection method, and the details of the accounts on each of the machines are indicated. The following is the list of machines:

Development server. Windows machine accessible through the VPN of the GEEDS group. As it is a Windows machine, the connection will be made through the Remote Desktop Connection tool, available on any Windows system. The IP address of this machine is 192.168.30.215, the user name of the account is administrator and the password is DeV.2021.

Test server. Linux machine used to deploy the versions of the application to be tested before moving them to production. The operating system of this machine is Ubuntu 20.04 and the connection is made through SSH. The name of the server where the machine is located is virtual.lab.inf.uva.es (port 20041), the username of the account is user and the password is cross.lab.inf.uva.es (port 20041). account is user and the password is crossroads2021. The back-end is available on port 20043 of the domain name, while the front-end is available on port 20042. As for the relevant directories on this machine, the code for each component is located in a subdirectory of the /home/user/git directory, while the front-end artifacts (after compiled) are located in the /var/www/html/crossroads-frontend directory and the apache server configuration is located in the /etc/apache2 directory.

Production server. Linux machine accessible through the GEEDS group's VPN that is used to deploy the is used to deploy the production versions of the application. The operating system of this machine is Debian 10 and the connection method is SSH. The IP address of this machine is 192.168.30.218, the account username is superadmin and the password is DIEA.2021. The public IP of this machine is 157.88.62.117 (domain name crossroads.geeds.eu). The front-end is available through port 80 of the above IP address, while the back-end is available through port 443. As for the relevant directories on this machine, the code for each component is located in a subdirectory of the /home/user/git directory, while the front-end artifacts (after compiled) are located in the /var/www/html/crossroads-frontend directory and the apache server configuration is located in the /etc/apache2 directory

4.8.1.1. FRONT END DEPLOYMENT COMMANDS

- Installation of project dependencies:

```
npm install
```

- Start the local development server (Spanish version):

```
ng serve -- configuration = es
```

- Start the local development server (English version):

```
ng serve -- configuration = en
```

- Compile the application to deploy it on the server:

```
npm run build
```

- Extract the file of static messages in English:

```
npm run i18n
```

- Restart the Apache server on which the front-end has been deployed.:

```
sudo systemctl restart apache2
```

- Check the status of the Apache server on which the front-end has been deployed:

```
sudo systemctl status apache2
```

- Open the graphical interface for running e2e tests:

```
npm run cypress
```

- Run e2e tests without the need for a graphical interface:

```
npx cypress run
```

4.8.1.2. FRONT END DEPLOYMENT GUIDE

To deploy the front-end code, first install the following dependencies:

- Node.js. Install version 12.20.2.
- NPM. Install version 8.5.5.
- Angular. Install version 13.1.3.

In order to perform the deployment, it is necessary to follow these steps:

1. Go to the project folder and run the npm install command. This will download all the modules on which the front-end depends. NOTE: if the project dependencies have not been updated, the execution of this step is skipped.
2. Run command npm run build This command generates in the dist folder of the project a folder with all the artifacts of the application, from the code itself to the resource files.
3. Delete the directory /var/www/html/crossroads-frontend. Copy the folder we have just generated into the /var/www/html directory.
4. Restart the Apache server (see command sheet). Optional: check the status of the web server, to make sure that everything is correct.

In the Git repository there is a continuous integration script that takes care of the automatic deployment to the production server automatically. This script, called .gitlab-ci.yml, will be executed each time a push of 1 or more commits is performed on the develop or master branches.

The internationalization of the front-end has been done by using the mechanisms provided by the Angular framework itself (package localize). This package makes use of different files in which the messages to be internationalized are indicated. To add new messages and/or update existing messages, it is necessary to execute the following steps:

a. Annotate the elements to be translated with the corresponding tags that mark that the text of that element is to be internationalized (see Angular documentation through this link). that element is going to be internationalized (see Angular documentation through this link).
b. Generate the file with the messages in the default language (in this case English): npm run i18n.
c. For all the new messages, make a copy of them in the message files of the other languages. the other languages.
d. For each of the copied messages, add the pair of target labels and add between the two labels the translated message in the corresponding language. the message translated into the corresponding language.
e. Optional: if the message is an update of a previous message, delete the old message from the language files. from the language files.

To run the e2e test suite of the application it is necessary to place the project folder as the current directory and execute the command: npm run cypress. This command opens a graphical interface in which all the files containing test cases of the application appear. If you click on each of the files, the tests associated with that file are executed. There is also an option to run all the tests of all the files. If you want to run the tests without the need for a graphical interface, run the command: npx cypress run.

To view the coverage analysis, you have to access the coverage/lcov-report directory that Cypress has created in the root directory of the project. The report is located in the index.html file.

4.8.2. BACK-END

Back-end deployment commands:

- Back-end (development server) start-up:

  mvn spring - boot : run

- Back-end (development) start-up (server):

  nohup mvn spring - boot : run &

- Data base reset (Linux):

  sh reset_db.sh

- Data base reset (Windows):

  reset_db . bat

- Execution of the script that generates the SQL with the translations of the form:

  python3 generate_form_translations_sql.py

- Execution of the script that loads data in MongoDB:

  python3 upload_patterns_and_graphics.pY

4.8.3. BACK-END DEPLOYMENT GUIDE

In order to start the application server, it is necessary to install the following dependencies:

- Java. Install JDK version 11.

- MySQL Server. Install the 8.0.x version. In addition, during the installation (or at the end of the installation) you have to set a password for the root user. This password must be "admin".
- MongoDB. Install version 4.4.1. As in the previous case, you must set a password for the root user (this password must be "admin" again).
- Maven. In the case of Maven, installing the last available version will be enough.

Once the dependencies are satisfied, the back-end can be started. To do so, the following steps must be executed:

1. Create in MySQL and MongoDB a database called crossroads.
2. Start the server (see command sheet). As this is the first time the server is started, all tables and collections will be created in the corresponding databases.
3. Populate the MySQL database with the information from the form, the hints and the moderators of the individual games. of the individual games. To do this, it is necessary to run a script, depending on the operating system of the machine on which it is run. operating system of the machine on which the deployment is to be performed. With this in mind, the db.bat script will be executed, if the OS is Windows, while in the case of Linux the reset db.sh script will be used. db.sh reset script will be used for Linux. The execution of each script will be carried out from the deploy subfolder of the project, making use of the commands indicated in section
4. Populate the Mongo database with the data from the simulations. To do this, two calls must be executed to the server that we have just set up. These two calls are uploadPatterns and uploadGraphics and require text files to be executed. The files are available through this link. To execute the two API calls, unzip the files in the deploy folder and execute the python script upload patterns and graphics.py (see command sheet). Once the two calls are executed with this script, all the data will be stored in their corresponding collections.

The above list lists the steps to be performed in order when performing the first back-end deployment. For subsequent deployments, only step 2 is required.

To update the version of this component that is deployed on the production server, the changes made in the master branch must first be integrated, since it is from this branch that the deployment must be performed. Considering this requirement, these are the necessary steps to complete the upgrade:

1. Connect via SSH to the machine. NOTE: to do this, it will be necessary to use the VPN of the GEEDS group. GEEDS group.
2. Access the /home/superadmin/git/backend directory.
3. Change the current branch to the master branch (command git checkout master).
4. Fetch the code changes from the remote repository (command git pull origin master).
5. Check if the back-end is running. To do this, run the ps -u superadmin command and check if in the list of running processes there is a process labeled java. If this process appears, it will be necessary to kill it by using the kill -9 command.
6. Optional: if the changes in the code imply a change in the MySQL database, it will be necessary to delete it and recreate it. database, it will be necessary to delete it and recreate it.
7. Start the server in the background (see command sheet).
8. Optional: if the changes in the code imply a change in the MySQL database, it will be necessary to repopulate the database. populate it again. To do this, it is necessary to perform step 3 described in the previous section.

With regard to internationalization, in order to use the script that creates the SQL that allows a new version of the form in another language to be entered into the database, it is necessary to create a text file containing the translations of the questions and answers. The structure of this file is as follows:

Language code. The first line of the file will contain the language code corresponding to the translations that are available in the file.

Translations of the questions. The next lines of the file will contain all the translations of the questions. These will be sorted in descending order by question number and each question will occupy a single line.

Separator character for questions and answers. Once the lines corresponding to the questions are finished, it is necessary to add the separator character $ (dollar sign) in a line, as this will indicate to the script that the questions are finished.

Translations of the answers. The last lines of the file will contain the translations of the answers, in descending order of option and question number.

In the deploy folder of the project there are two examples of files with this structure (files crossroads en.txt and crossroads en.txt). Once the text file is built, a couple of modifications need to be made to the Python script before it can be used. First, the FILES variable needs to be modified to indicate the name of the file(s) to be used to generate the SQL for the new translations. Secondly, it is necessary to update the values of the variables id question translation and id answer translation, because these variables indicate the initial value of the translations identifiers, so the value they should take is equal to the current final id + 1.

Finally, and in order to fully integrate the version with the new language, the LocaleConfig class of the LocaleConfig.java file (package config) must be modified to include the new language code supported by the app.

To add a new version of the internationalized messages returned by the back-end, it is necessary to create in the resources folder of the project a new file called messages <codec-language>.properties, copy all the lines that appear in the messages.properties file and translate the part that is to the right of the = symbol, respecting the left part and the marks of the parameters that the messages can receive (symbols { and }) and content of the messages). In addition, as a requirement to be able to use these files it will be necessary to have applied the modification to the LocaleConfig class mentioned at the end of the previous section.

To add a new internationalized message to the back-end, it is necessary to include a new line in all the message files (resources folder). This new line will consist of the following components: An identifier (to be used to retrieve the message when necessary) followed by the = sign and the string value. If the string has some parameters, these will be indicated by the symbols { and }, between which the position of each argument will be indicated, starting with 0. An example of what a message would look like is shown below:

example.player=Player {0} has id: {1}

To use the newly added message it will be necessary that the Java class that is going to make use of that message declares an attribute of type LocalizedMessagesResolver and that the corresponding methods of that attribute are invoked. corresponding to this attribute. The documentation of the methods of this class can be found in their own code.

4.8.4. RECOMMENDER

4.8.4.1. RECOMMENDER DEPLOYMENT COMMANDS

- Creation of the image Docker with the recommender:

  sudo docker build -- tag recomendador .

- Start-up of the recommender:

  sudo docker run -p 8000:5000 recomendador

- Start-up of the recommender (in background):

  sudo docker run -d -p 8000:5000 recomendador

- Checking the docker images that are running:

  sudo docker ps

- Destruction of all active containers.:

  sudo docker kill $ (sudo docker ps -q)

- Generation of the catalog of messages to be internationalized:

  pybabel extract -F babel . cfg -k lazy_gettext -o messages . pot .

- Message file generation for a specific language:

  pybab el init -i messages . pot -d translations -l < codigo - idioma >

- Compile all translation files:

  pybabel compile -d translations

4.8.4.2. RECOMMEDER DEPLOYMENT GUIDE

To deploy this service, it is only necessary to have docker and python version 3.8 installed. The rest of the dependencies will be installed during the process of creating the docker image that will contain this service.

In order to deploy the component on a Linux machine, it is necessary to follow the following steps:

1. Run the command sudo docker build --tag recommender, to create a docker image named recommender.
2. Once the docker image is created, use the command sudo docker run -p 8000:5000 recommender, where 8000 is the port of the machine to which the requests will be made and 5000 is the port on on which the docker server is running. To run the above command in the background, the -d option must be added.
3. To stop the background execution of the recommender, the command sudo docker kill $(sudo docker ps -q) must be executed.

The recommender is internationalized using the Flask-Babel library. This library makes use of a set of files, which act as a catalog of messages for a given language. In this way, and thanks to the content of the Accept-Language header of the HTTP requests received, the library will use the messages of the file corresponding to the language requested by the user in the request header.

The list of steps that are necessary to create and maintain these message files is shown below:

1. Annotate all the texts to be translated in the Python files with the gettext or lazy gettext tags (see documentation of both tags to know when to use one or the other).

2. Generate the base file from which the translations will be generated: pybabel extract -F babel.cfg -k lazy gettext -o messages.pot.
3. Optional: if the language for which you want to generate the messages file is new, you have to execute the following command: pybabel init -i messages.pot -d translations -l <language-code>.
4. Edit the file translations/<language-code>/LC MESSAGES/messages.po to include the message translations.
5. Compile all the translations: pybabel compile -d translations.
6. Optional: to update a message you have to modify this message in the code, execute step 2 and execute the following command so that Babel includes the new message in the translation's files: pybabel update -i messages.pot -d translations. From this point on, steps 4 and 5 must be carried out.

If you want to add a new language to the application, in addition to the previous steps that serve to generate the file for that language, it is necessary to edit the variable SUPPORTED LANGUAGES of the code file app.py, adding to the list the code of the new language to be supported by the application py code file, adding to the list the code of the new language to be supported by the application.

5. MODEL EXPLORER

The Model Explorer is a user-friendly and easy-to-use web-based application to explore WILIAM IAM without needing to be a modelling expert. With a reduced functionality, it main goal is to increase the awareness in the civil society in terms of climate change, helping in the definition of more sustainable future scenarios. The developed application will make possible to change specific parameters to configure your own scenario based on the pre-configured scenarios of WILIAM model. The tool will offer two modes of use: as a registered user or without register. Without registering to the tool, you can configure and explore different scenarios from the ones configured by the LOCOMOTION project. These scenarios can only be saved if you are registered with the tool.

5.1. IDENTIFIED REQUIREMENTS

Considering the scope of the tool and the user requirements, a set of functional and non-functional requirements for the Model Explorer were developed. The list of functional requirements that will be addressed by the Model Explorer tool is presented in the Table 1. Functional requirements are grouped by Use Case. This table also includes an ID for requirement identification and the description of the Functional Requirement. The non-functional requirements of the tool related to the performance and security of the information provided by the user and applied in the different assessment steps are included in Table 2.

Table 1. Model Explorer functional requirements.

Use case name		Req. ID	Functional Requirement description
Registration	Any user	FR 1.1	The Model Explorer tool should allow any user (citizens and experts) to access to the tool without create an account.
	Registered users	FR 1.2.1	The Model Explorer tool should allow any user (citizens and experts) to register into the pre-design tool providing basic information (e.g. email, login, password or other additional information as position).
		FR 1.2.2	The Model Explorer tool should allow any registered user to log into the pre-design tool by inserting a login and a password.
		FR 1.2.3	The Model Explorer tool should allow any registered user to modify their account credentials provided to the pre-design tool.
Project		FR 2.1	The Model Explorer tool should allow any registered user to create a new project.

	Project management for registered users	FR 2.2	The Model Explorer tool should allow any registered user to delete a new project.
		FR 2.3	The Model Explorer tool should allow any registered user to modify the previous developed projects.
Assessment	Any user	FR 3.1	The Model Explorer tool should allow the user to visualize the assumptions included during the Model implementation by means of a suitable explanation of each assumption.
		FR 3.2	The Model Explorer tool should allow the user to compare pre-defined scenarios (BAU, SSPs, Green growth) included in the tool.
		FR 3.3	The Model Explorer tool should allow the user to create their own scenario selecting the most suitable hypothesis and policies, and compare it respect to the pre-defined scenarios included in the tool.
		FR 3.4	The Model Explorer tool should allow the user to obtain an explanation of each policy during scenario development to select the most suitable policies according to the scenario requirements.
		FR 3.5	The Model Explorer tool should allow the user to visualize the results of the created scenario by means of relevant graphs.
		FR 3.6	The Model Explorer tool should allow the user to select different graphs to visualize the final results of the created scenario.
		FR 3.7	The Model Explorer tool should allow the user to visualize a summary of the selected policies and the impacts of their implementation
Export scenario	For registered users	FR 4.1	The Model Explorer tool should allow the user to export the results of the scenario including all the policies choose for it development

Table 2. Model Explorer non-functional requirements.

Name	NFR1: Performance requirements
Description	The systems should provide a sufficiently high availability and be scalable as well as fault-resilient. Efficiency, computational capacity, reliance, robustness and replicability concepts are considered. • Reliability. Real time simulations and results storage need a high quantity of time for computing and a big amount of data to be stored. A malfunction of the system could produce stops, the loss of results and the loss of time. • Response time: The system should be able to respond in a limited time in order to allow a fluent activity in the stages of interaction with the user. • Computational capacity. The Model Explorer should have enough computational capacity to run simulations with the set of pre-selected policies. • Scalability: This issue needs to be covered to ensure the implementation of new functionalities for the tool in an easy way. We will provide documentation on repositories, code and function to ensure the tool comprehension. • Standard protocols. The platform should use standard protocols on code development and data integration to ensure a success deployment.
	Standard
	Issues related with computational capacity are key to ensure the tool success.
Name	**NFR2: Security requirements**
	The systems should be able to ensure confidentiality and integrity of collected data from users (personal data) and location data provided to create and optimize the district heating network. • Authentication and authorisation: The tool should provide appropriate interfaces to support access, registration and password recovery to users. • Integrity: The system should provide meta-information about origin and trust of the gathered data and protect the data against malicious of accidental modification. • Confidentiality: The tool should ensure the confidentiality of information about developed projects and sensitive information provided by the user.
	High
	Security requirements are mandatory and needed for operating the tool

The collected requirements are translated into use cases to obtain a general idea on how the tool will run. Table 3 presents the list of use cases that are deeply described in section 6.1.1 (Table 4 to Table 12). These use cases, present the interactions between the user and the tool, providing also an insight on the sequence to be followed and the expected outcomes in the use of the tool.

Table 3. Model Explorer use cases.

Use case number	Use case name
UC0	Assessment using the tool
UC1	Register as tool user
UC2	Log into the tool
UC3	Recover password
UC4	Create new policy scenario
UC5	Review saved scenarios
UC6	Export chosen scenario
UC7	Modify user data
UC8	Visualize model assumptions

The main attributes associated to each use case definition are:

- ***Description:*** A sentence explaining the main use case purpose.
- ***Actors***: who interacts with the tool in each use case.
- ***Inputs***: actions, files or data needed to fulfil each case study.
- ***Outputs:*** actions, data, results or others resulting from the use case implementation.
- ***Tools***: external or internal tools or modules that are needed for the use case execution.
- ***Preconditions***: previous conditions that must be satisfied by the tool.
- ***Normal sequence:*** progression of tasks to be followed by the user to complete the use case.
- ***Post condition***: what must be fulfilled for the use case to be finalised.
- ***Exception***: identified issues that may appear if the interaction with the tool is not as expected.
- ***Linked Use Cases***: other use cases directly related to the one at issue.
- ***Importance***: relevance of the use case compared to others to fulfil the whole tool objective.
- ***Urgency:*** priority for the implementation.
- ***Comments:*** additional comment for use case definition.

5.1.1. DETAILED USE CASES

Table 4. Use Case 0 (UC0): Assessment using the tool.

Use case	**UC0 - Assessment using the tool**
Identifier	**Assessment**
Description	Start a new assessment using the Model Explorer tool
Actors	All tool users (owner)
Inputs	Tool access by means of the link included on the LOCOMOTION website
Outputs	Tool access by means of two alternatives: registered user and user without registration
Tools	No external tool is required
Preconditions	The user needs to have an explanation of the two alternatives that are offered by the tool
Normal sequence	1.- The user visits the Model Explorer tool Homepage section in the website
	2.1- If the user clicks on the "Access without registration" button, the tool redirect the user to start the development of a new scenario assessment without register in the tool (UC4)
	2.2- If the user clicks on the "Register" button, the tool redirect the user to create an account that will make possible to save developed scenarios
	3.- The use case ends
Post condition	User is redirected to the selected tool mode
Exception	-
Linked Use Cases	UC1 - Register as tool user; UC2 - Log into the too; UC4 - Create new policy scenario
Importance	High
Urgency	Non-urgency
Comments	-

Table 5. Use Case 1 (UC1): Register as a tool user.

Use case	**UC1 - Register as a tool user**
Identifier	**Register**
Description	Register through a user account that has been created using an email and a password
Actors	All tool users (owner)

Inputs	Information of the users - Email address - Password
Outputs	Tool access by means of the data user account stored into the user data repository
Tools	No external tool is required. The Project Database can also be used to this aim
Preconditions	The user has to have an email account
	The user has to select a non-repeated password
Normal sequence	1.- The user visits the Model Explorer tool Homepage section in the website
	2.- The user clicks on the "Register" button in order to create an account
	3.- The system displays a form that includes the space to input the register credentials
	4.- The user enters the email/user name and password
	5.- The system validates email and password
	6.- The user is registered in the tool
	7.- The use case ends
Post condition	User registered in the tool successfully
Exception	1.1.- If the email address or password is invalid
	1.2.- The system will display an error message and prompt for retyping the email address and password
Linked Use Cases	UC0 - Assessment using the tool
Importance	High
Urgency	Non-urgency
Comments	-

Table 6. Use Case 2 (UC2): Log into the tool.

Use case	**UC2 - Log into the tool**
Identifier	**Login**
Description	Log into the tool through a user that has been created using an email and a password
Actors	All registered users (owner)
Inputs	Information of the users - Email address - Password
Outputs	Tool access by means of the data stored into the user data repository
Tools	No external tool is required. The Project Database can also be used to this aim
Preconditions	The user has to be registered in the platform
	The user must not be already logged in the tool
Normal sequence	1.- The user visits the Model Explorer tool Homepage in the project website
	2.- The user clicks on the "Login" button in order to log in to the online tool
	3.- The system displays a form to the user to input login credentials
	4.- The user enters the email/user name and password
	5.- The system checks the authentication for user's information and validates
	6.- The user is logged in the tool
	7.- The use case ends
Post condition	User logged in the tool successfully
Exception	2.1.- If the email address or password is invalid
	2.2.- The system will display an error message and prompt for retyping the email address and password
Linked Use Cases	UC1 – User registration
Importance	High
Urgency	Non-urgency
Comments	-

Table 7. Use Case 3 (UC3): Recover password.

Use case	**UC3 - Recover password**
Identifier	**Recover**
Description	Recover a password for registered users by means of the provided email during the registration process
Actors	All registered users (owner)

Inputs	Information of the users: - Email address - Password
Outputs	Tool access by means of a new password stored in the user data repository
Tools	No external tool is required. The Project Database can also be used to this aim
Preconditions	The user has to be registered in the platform
	The user must not be already logged in the tool
Normal sequence	1.- The user visits the Model Explorer tool Homepage in the project website
	2.- The user clicks on the "Login" button in order to log in to the online tool
	3.- The user clicks on "Forgot password" to recover the password
	4.- The system displays a form to the user to include the new credentials
	5.- The user enters the new password
	6.- The system checks the authentication for user's information and validates
	7.- The user is logged in the tool
	8.- The use case ends
Post condition	User logged in the tool successfully
Exception	2.1.- The password is invalid or you are not a registered user
	2.2.- The system will display an error message and prompt for retyping the email address and password
Linked Use Cases	UC2 – Log into the tool
Importance	Normal
Urgency	Non-urgency
Comments	-

Table 8. Use Case 4 (UC4): Create new policy scenario.

Use case	**UC4 - Create new policy scenario**
Identifier	**NewScenario**
Description	All the Model Explorer users have to be able to create a new policy scenario for its assessment
Actors	Experts and other users (e.g. citizens, policy-makers)
Inputs	Data stored into the data repository
Outputs	New Custom policy scenario to be stored in the project repository
Tools	No external tools are required for this use case
Preconditions	The Model Explorer tool has to work properly
	Project database has to be functional
Normal sequence	1.- The user selects in the dropdown the "Custom scenario" option
	2.- The system displays the "Custom scenario" values (zero) for all the variables that the user could modify in the model
	3.1.- The user selects using dropdowns and bottoms the most suitable policies and hypothesis considering their knowledge respect to climate change
	3.2.- The user clicks on "View policy" to review a brief description of each policy
	4.- The Model Explorer tool automatically update the scenario defined by the user
	5.- The user clicks on "Safe scenario" to store the developed scenario in the database
	6.- The user specifies the name of this "Custom scenario"
	7.- The system safes the developed scenario
	8.- The use case ends
Post condition	The scenario is saved according to the criteria defined by the users.
Exception	-
Linked Use Cases	UC2-Log into the tool
Importance	Normal
Urgency	Non-Urgent
Comments	-

Table 9. Use Case 5 (UC5): Review saved scenarios.

Use case	**UC5 - Review saved scenarios**
Identifier	**ReviewScenarios**
Description	The users must be able to review the saved scenarios developed using the Model Explorer tool

Actors	Experts and other tool users
Inputs	Custom policy scenarios stored in the database
	Other necessary data included in the database
Outputs	Result of the selection of scenarios including graphs and a summary of policies
Tools	Model Explorer tool and the project database
Preconditions	The Model Explorer tool has to work properly
	Project database has to be functional
Normal sequence	1.- The user clicks on "Saved scenarios" button
	2.- The system displays the results of the saved scenarios
	3.- The user clicks on each scenario to visualize it
	4.- The system presents the scenario including graphs and a summary of the implemented policies
	5.- The use case ends
Post condition	Selected scenario results are presented to the user
Exception	-
Linked Use Cases	UC3 - Create new policy scenario
Importance	Normal
Urgency	Non-urgent
Comments	-

Table 10. Use Case 6 (UC6): Export chosen scenario.

Use case	**UC6 - Export chosen scenario**
Identifier	**ExportResults**
Description	The users must be able to modify password, user name and email accessing to the user area that is included in the tool
Actors	All registered users (owner)
Inputs	User data stored into the data repository of the tool
Outputs	New user name, email or password stored in the tool
Tools	Model Explorer tool and the project database
Preconditions	The user has to be registered in the platform
	The user must not be already logged in the tool
Normal sequence	1.- The user selects one of the stored Custom policy scenarios
	2.- The user clicks on "Export scenario" button
	3.- The system displays the results of the selected scenarios
	4.- The user clicks on "Export results" button and specifies the save as type
	5.- The system prompts the document type and execute the export procedure
	6.- The use case ends
Post condition	Selected scenario results are exported as the selected file type
Exception	-
Linked Use Cases	-
Importance	Normal
Urgency	Non-urgent
Comments	-

Table 11. Use Case 7 (UC7): Modify user data.

Use case	**UC7 - Modify user data**
Identifier	**Modifydata**
Description	The users must be able to export the results which are obtained from the assessment stages through the Model Explorer tool
Actors	Experts and other tool users
Inputs	Custom policy scenarios developed by the user and stored in the database
	Other necessary data included in the database
Outputs	Results of the Custom policy scenario including graphs and a summary of policies
Tools	Model Explorer tool and the project database
Preconditions	The Model Explorer tool has to work properly
	Project database has to be functional
Normal sequence	1.- The user clicks on "User area" button
	2.- The system displays the "User area" section

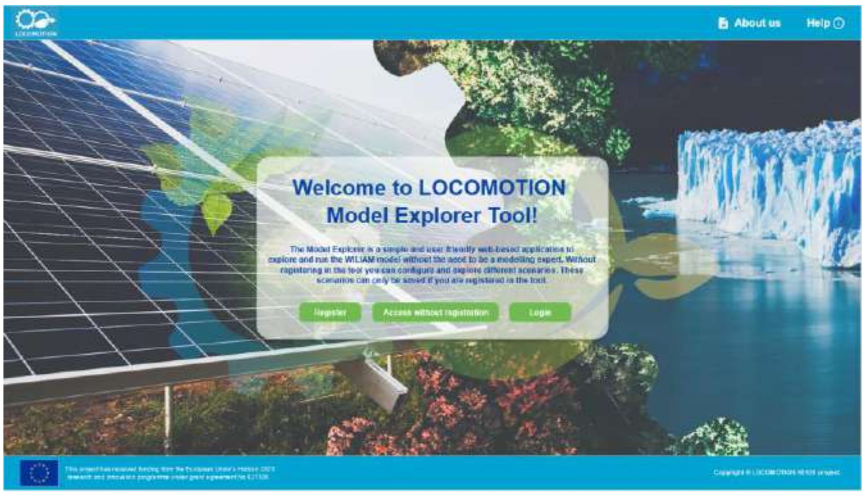
	3.- The user clicks on the aspect that want to change
	4.- The system starts edition mode
	4.- The user modifies the attribute a save the changes
	6.- The use case ends
Post condition	User stored data are modified
Exception	-
Linked Use Cases	UC4 - Review saved scenarios
Importance	Normal
Urgency	Non-urgent
Comments	-

Table 12. Use Case 8 (UC8): Visualize model fixed policies.

Use case	**UC8 - Visualize model fixed policies**
Identifier	**FixedPolicies**
Description	The users must be able to visualize all the model parameters that are stablish in a constant value during the model simulation
Actors	All tool users (owner)
Inputs	Scenario variables stored in the data repository of the tool
Outputs	Access to visualize constant variables in the simulation
Tools	No external tool is required
Preconditions	The user must be logged or not in the tool
	The user needs to have access to the policy scenario view
Normal sequence	1.- The user clicks on "Model assumptions"
	2.- The system displays different groups of variables
	3.- The user clicks on each group and variable to check the value used for simulation
	4.- The user clicks on another variable to change the visualized parameter
	5.- The use case ends
Post condition	Selected scenario results are exported as the selected file type
Exception	-
Linked Use Cases	UC4 - Create new policy scenario
Importance	Normal
Urgency	Non-urgent
Comments	-

5.2. MOCKUPS

This section, present the initial definition and appearance of the Model Explorer mockups. The initial mockups were developed using a combination of Microsoft PowerPoint and moqups application. A separate slide was created for each of the pages of the tool in order to collect the main steps in the use of the tool. The mockup results, are shown in Figure 94 to Figure 105:



Welcome to LOCOMOTION Model Explorer Tool!

The Model Explorer is a simple and user friendly web-based application to explore and run the WILIAM model without the need to be a modelling expert. Without registering in the tool you can configure and explore different scenarios. These scenarios can only be saved if you are registered in the tool.

Figure 94: Welcome page.

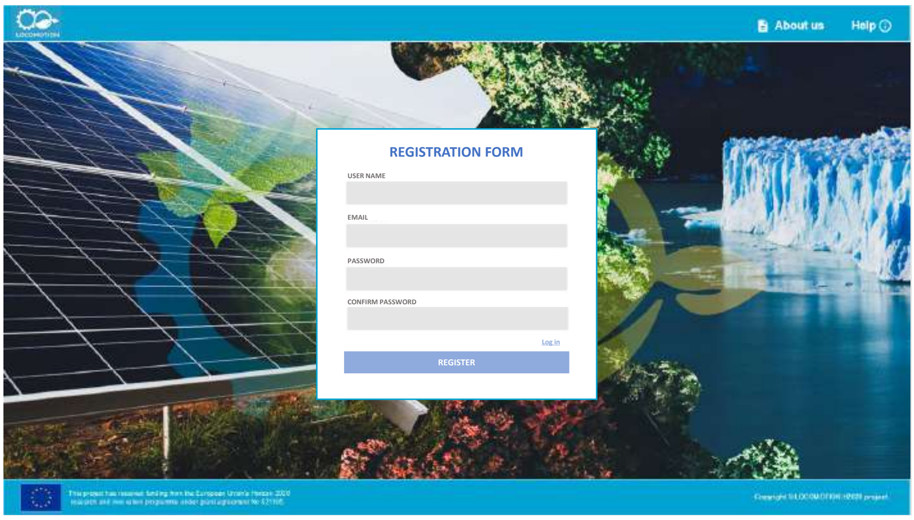

About us
Help
REGISTRATION FORM
USER NAME
EMAIL
PASSWORD
CONFIRM PASSWORD
Log in
REGISTER
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821105
Copyright © LOCOMOTION H2020 project

Figure 95: Registration page.

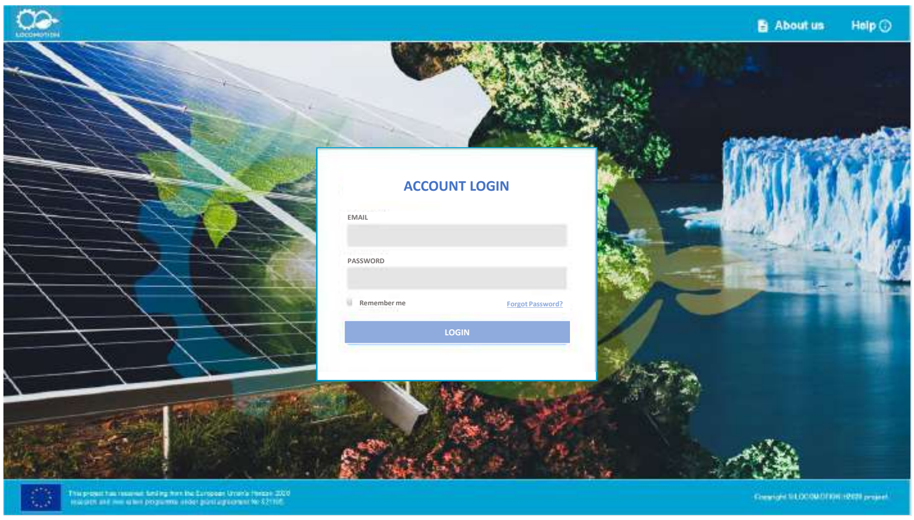

About us
Help
ACCOUNT LOGIN
EMAIL
PASSWORD
Remember me
Forgot Password?
LOGIN
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821105
Copyright © LOCOMOTION H2020 project

Figure 96: Login page.

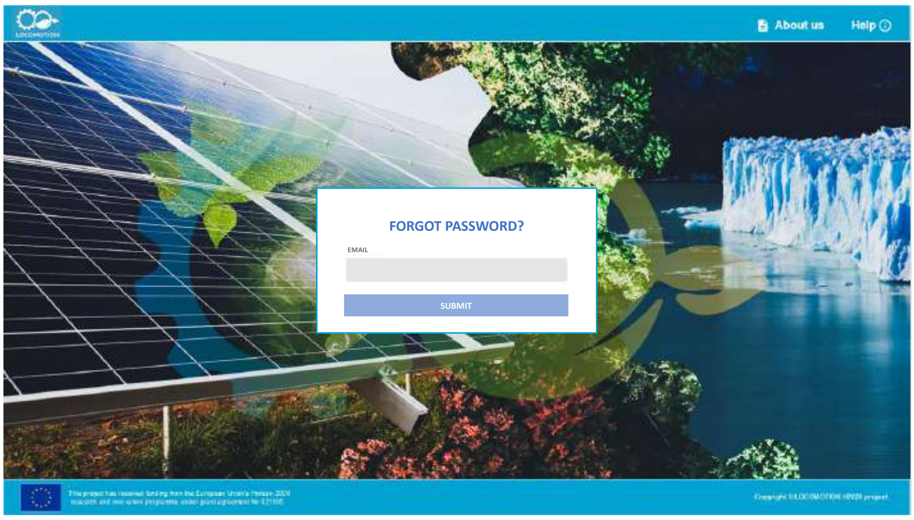
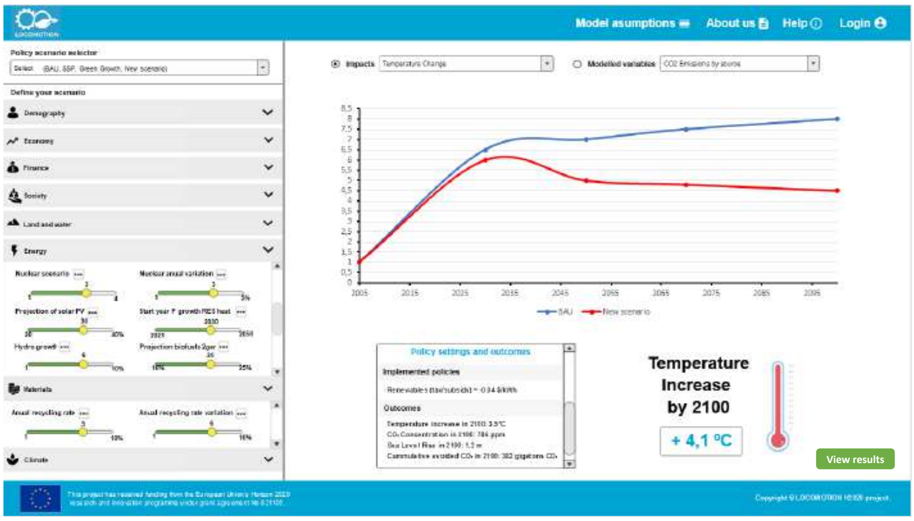

About us
Help
FORGOT PASSWORD?
EMAIL
SUBMIT
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821105
Copyright © LOCOMOTION H2020 project

Figure 97: Password recovery page.


Policy settings and outcomes

Implemented policies

Renewables (tax/subsidy) = 0.34 $/kWh

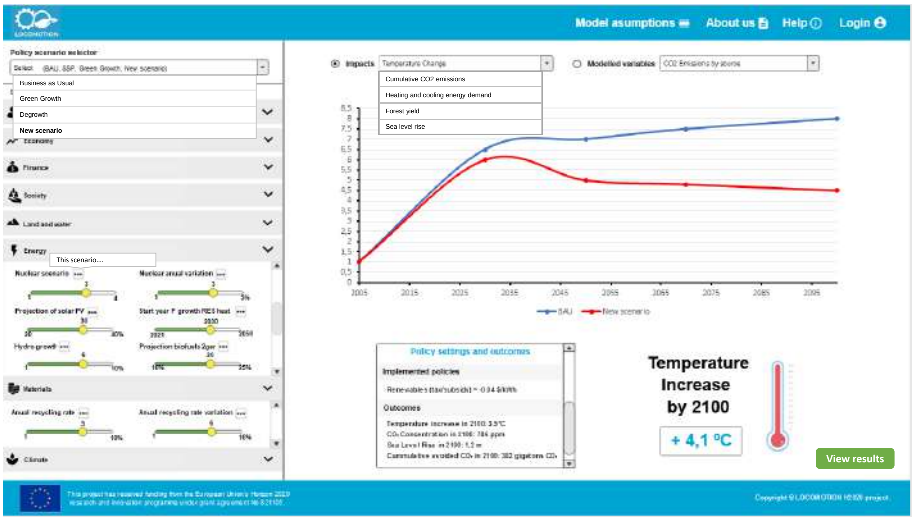
Outcomes

Temperature increase in 2100: 3.5°C
CO₂ Concentration in 2100: 784 ppm
Sea Level Rise in 2100: 1.2 m
Cummulative avoided CO₂ in 2100: 382 gigatons CO₂

Temperature Increase by 2100

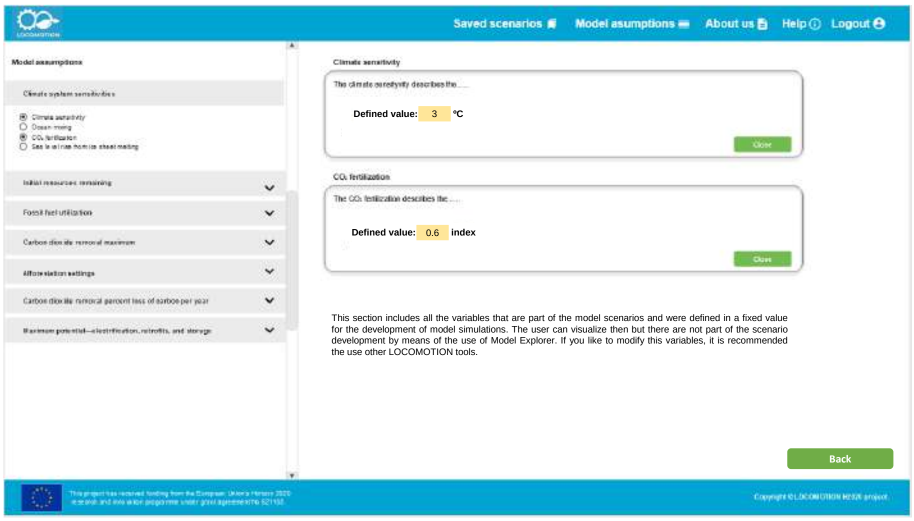
+ 4,1 ºC

Figure 98: Policy action scenario implementation.


Business as Usual
Green Growth
Degrowth
New scenario

Cumulative CO2 emissions
Heating and cooling energy demand
Forest yield
Sea level rise

Figure 99: Different dropdown options for policy action scenario implementation.


Climate sensitivity

The climate sensitivity describes the .....

Defined value: 3 ºC

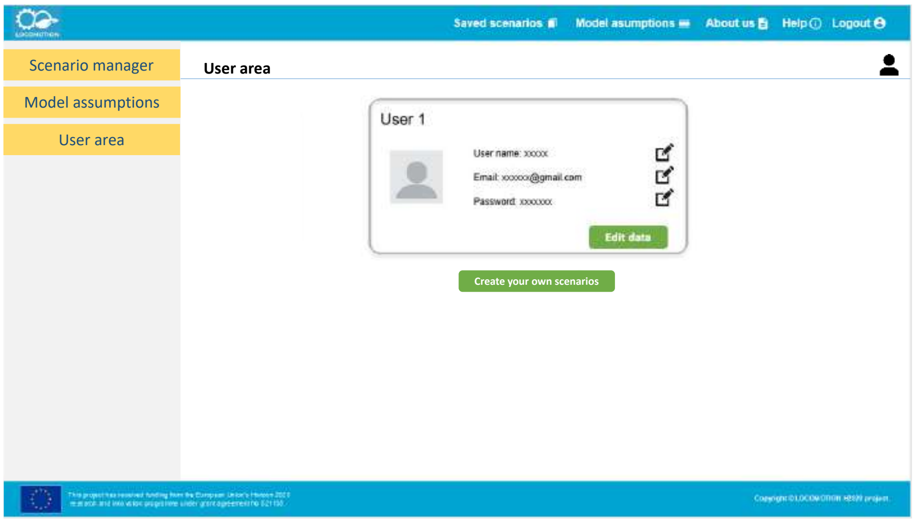
CO₂ fertilization

The CO₂ fertilization describes the .....

Defined value: 0.6 index

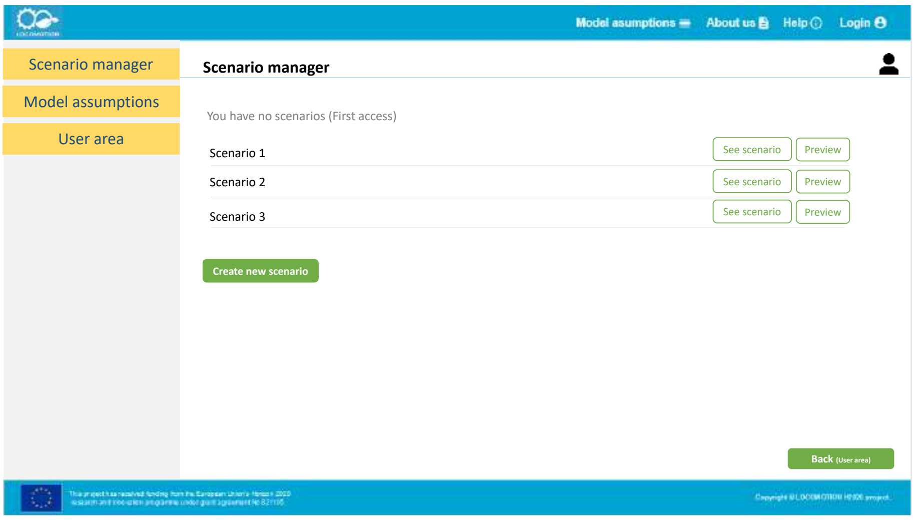
This section includes all the variables that are part of the model scenarios and were defined in a fixed value for the development of model simulations. The user can visualize then but there are not part of the scenario development by means of the use of Model Explorer. If you like to modify this variables, it is recommended the use other LOCOMOTION tools.

Figure 100: Model assumptions view.



User name: xxxxx
Email: xxxxxx@gmail.com
Password: xxxxxxx

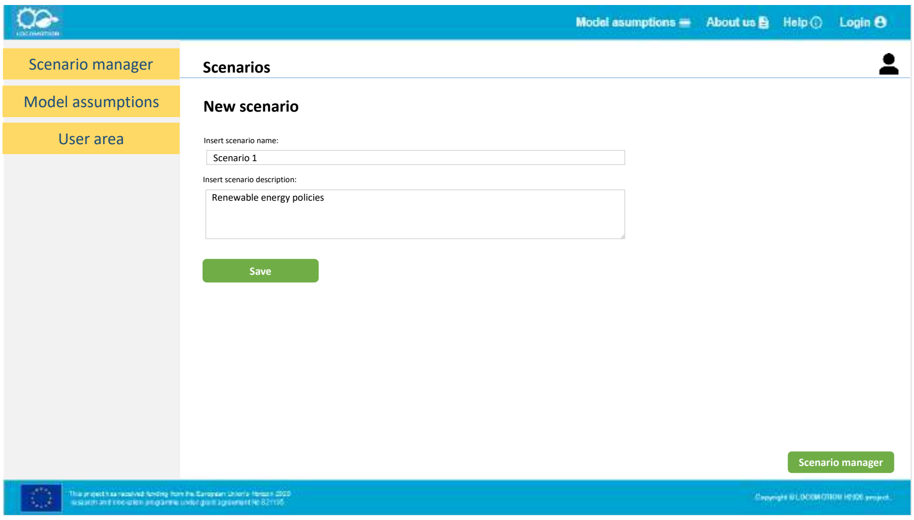
Figure 101: Modify user name and password page.



You have no scenarios (First access)

Scenario 1
Scenario 2
Scenario 3

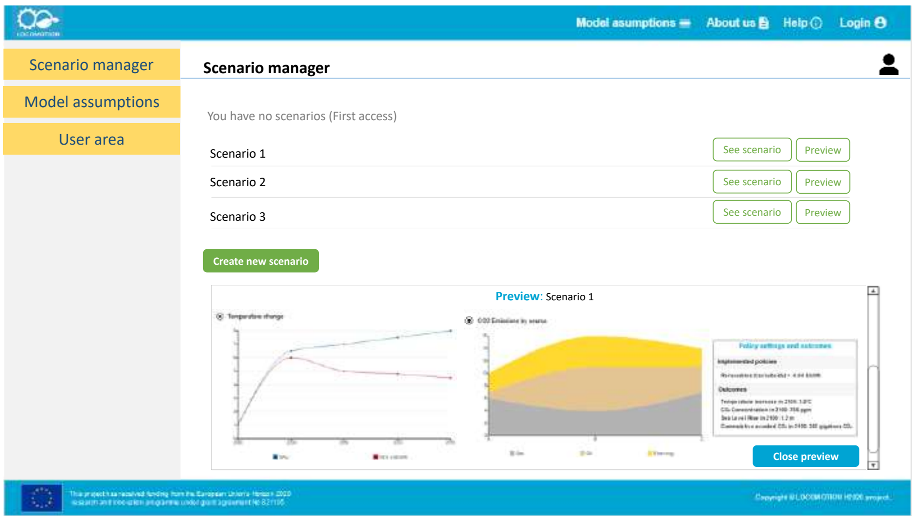
Figure 102: Scenario manager.



Insert scenario name: Scenario 1

Insert scenario description: Renewable energy policies

Figure 103: Create new policy action scenario.



Scenario manager

You have no scenarios (First access)

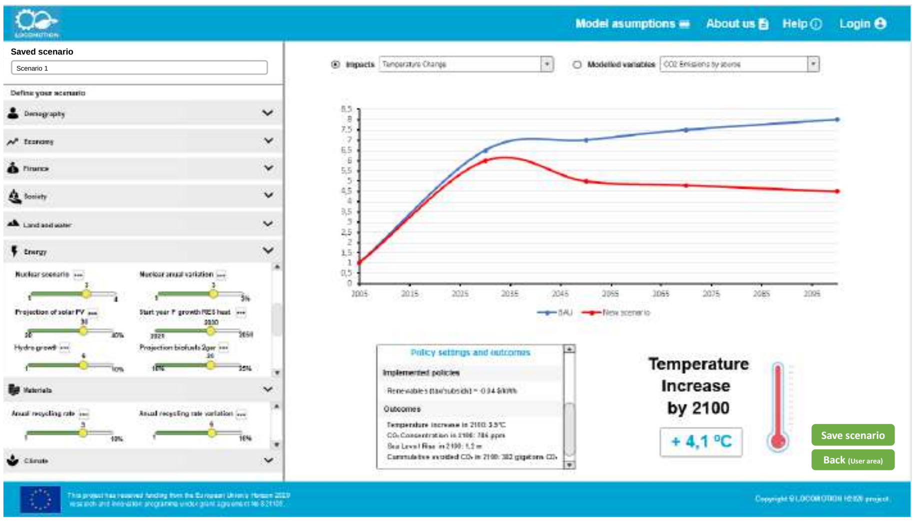
Scenario 1

Scenario 2

Scenario 3

Preview: Scenario 1

Figure 104: Pre-visualization of saved policy action scenarios.



Temperature Increase by 2100

+ 4,1 ºC

Figure 105: View saved policy action scenario.

5.3. GUI DESIGN

Based on the generated mockups, the design was created by Ondeuev using the InVision platform (https://www.invisionapp.com/). As it was explained before, this task was carried out based on the Graphical Identity Manual of the project, with the objective of maintaining a uniform design across the three LOCOMOTION applications. Figure 106, presents the design of all the interfaces that are part of the Model Explorer tool. The four main interfaces of the tool are detailed in a big size in Figure 107 and Figure 108.

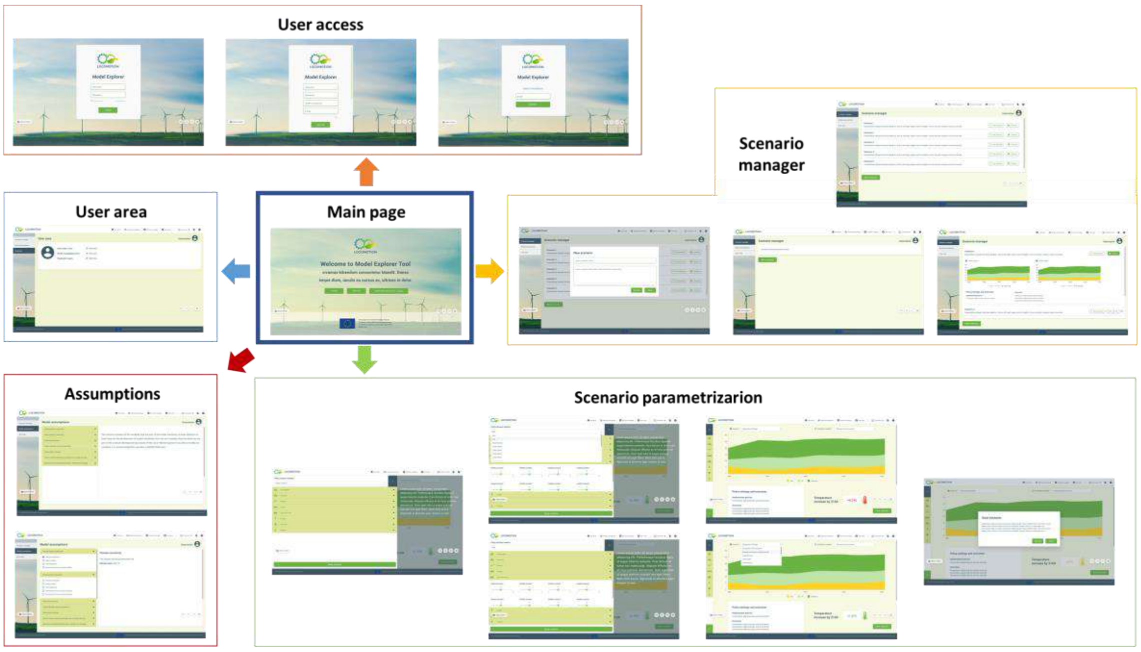


Figure 106: General view of Model Explorer interfaces design.

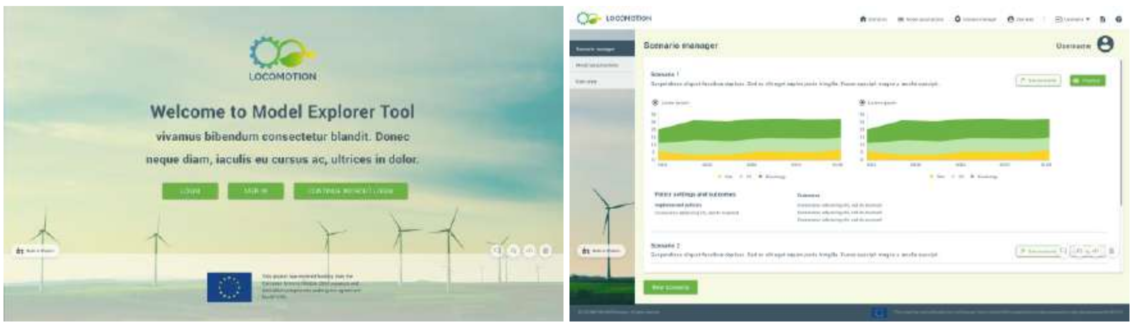

LOCOMOTION
Welcome to Model Explorer Tool
vivamus bibendum consectetur blandit. Donec neque diam, iaculis eu cursus ac, ultrices in dolor.
Scenario manager
Username

Figure 107: Main page interface (left) and Scenario manager interface (right) of the Model Explorer tool.

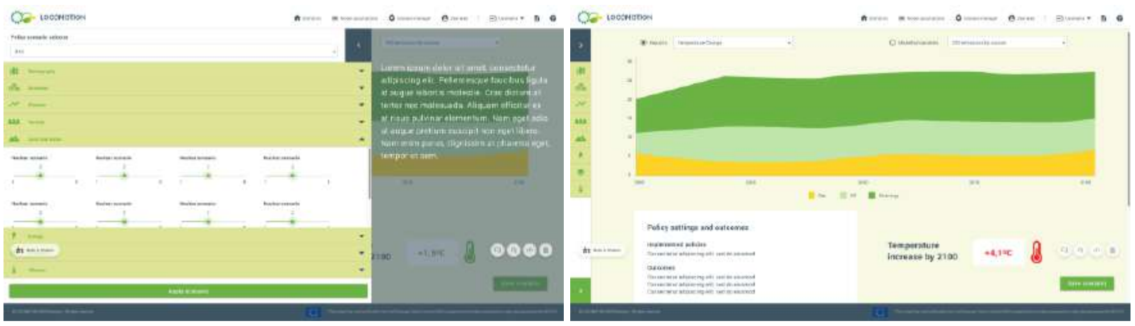


Figure 108: Policy implementation interface (left) and Results view interface (right) of the Model Explorer tool.

5.4. SELECTED TECHNOLOGIES

The procedures of work, as they are reflected in the Architecture Diagram (Figure 1) can be divided into three related branches: the security branch, the service branch and the generation branch. The security branch is related to the creation, management and permissions of users. It uses the following technologies: (i) Python FastAPI to build the API and the REST API, (ii) Keycloack as user manager, and (iii) PostgreSQL for the database.

The service branch includes the procedures to make the online tool work. And that is done with the REST API and PostgreSQL, that are shared with the security branch. In parallel, the generation branch deals with the creation of scenarios data that could be presented through the visualization of graphics from the tool. It uses the following technologies: (i) REDIS database management system and, (ii) Celery. This branch also uses Python FastAPI shared with the other branches, for the API connection with the application, in case of requesting a simulation.

Moreover, and related to the three branches, it can be considered the online tool itself, that has been developed with the following technologies: JavaScript and React. Each of the technologies used will be described below.

- **Fast API**

FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.7+ based on standard Python type hints. The key features are:

- Fast: Very high performance, on par with NodeJS and Go (thanks to Starlette and Pydantic). One of the fastest Python frameworks available.
- Fast to code: Increase the speed to develop features by about 200% to 300%.
- Fewer bugs: Reduce about 40% of human (developer) induced errors.
- Intuitive: Great editor support. Completion everywhere. Less time debugging.
- Easy: Designed to be easy to use and learn. Less time reading docs.
- Short: Minimize code duplication. Multiple features from each parameter declaration. Fewer bugs.
- Robust: Get production-ready code. With automatic interactive documentation.
- Standards-based: Based on (and fully compatible with) the open standards for APIs: OpenAPI (previously known as Swagger) and JSON Schema.

This technology has been utilized to create the API REST that works as interface between the online tool and the PostgreSQL database, the API REST that contacts the Celery manager of users, and the API REST that ties the REDIS database system inside the Vensim server with the tool.

- **Keycloack**

It is a tool for “Identity and Access Management”, as written on their project page on GitHub. Additionally, Keycloak is an open-source tool currently licensed with Apache License 2.0. It is also an upstream project for Red Hat SSO, in terms of handling an enterprise solution. The full list of supported platforms depends on the used protocol, currently Keycloak supports three different protocols, and can be viewed in documentation. Keycloak’s initial release took place in September 2014; the current version is 15.0.2 (2021-10-09). It is developed and maintained by people from Red Hat. Keycloak main Features:

- Multiple Protocols Support. As for now Keycloak supports three different protocols, namely - OpenID Connect, OAuth 2.0 and SAML 2.0.
- SSO. Keycloak has full support for Single Sign-On and Single Sign-Out.
- Admin Console. Keycloak offers web-based GUI where you can “click out” all configurations required by your instance to work as you desire.
- User Identity and Accesses. Keycloak can be used as a standalone user identity and access manager by allowing us to create user’s database with custom roles and groups. This information can be further used to authenticate users within our application and secure parts of it based on pre-defined roles.
- External Identity Source Sync. In case when your client currently has some type of user database, Keycloak allows us to synchronize with such database. By default, it supports LDAP and Active Directory but you can create custom extensions for any user database using Keycloak User storage API. Keep in mind that such a solution may not have all data necessary for Keycloak to be fully functional, so remember to check if your desired functionality works.

- Identity Brokering. Keycloak can also work as a proxy between your users and some external identity provider or providers. Their list can be edited from Keycloak Admin Panel.
- Social Identity Providers. Additionally, Keycloak allows us to use Social Identity Providers. It has built-in support Google, Twitter, Facebook, Stack Overflow but, in the end, you have to configure all of them manually from admin panel. The full list of supported social identity providers and their configuration manual can be found in Keycloak documentation.
- Pages Customization. Keycloak lets you customize all pages displayed by it to your users. Those pages are in .ftl format so you can use classic HTML markups and CSS styles to make the page fit your application style and your company brand. You can even put custom JS scripts as part of pages customization so possibilities are limitless.

This tool has been deployed in order to manage the security features related to users.

- **PostgreSQL**

PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. The origins of PostgreSQL date back to 1986 as part of the POSTGRES project at the University of California at Berkeley and has more than 35 years of active development on the core platform.

PostgreSQL has earned a strong reputation for its proven architecture, reliability, data integrity, robust feature set, extensibility, and the dedication of the open source community behind the software to consistently deliver performant and innovative solutions. PostgreSQL runs on all major operating systems, has been ACID-compliant since 2001, and has powerful add-ons such as the popular PostGIS geospatial database extender. It is no surprise that PostgreSQL has become the open source relational database of choice for many people and organisations. It includes de following features:

- Data Types: Primitives, like Integer, Numeric, String, Boolean. Structured like Date/Time, Array, Range / Multirange, UUID. Document: including JSON/JSONB, XML, Key-value (Hstore). Geometry, like Point, Line, Circle, Polygon. And customizations, in the form of Composite, Custom Types
- Data Integrity: UNIQUE, NOT NULL, Primary Keys, Foreign Keys, Exclusion Constraints, Explicit Locks, Advisory Locks.
- Concurrency, Performance: Indexing: B-tree, Multicolumn, Expressions. Partial Advanced Indexing as GiST, SP-Gist, KNN Gist, GIN, BRIN, Covering indexes, Bloom filters. Sophisticated query planner / optimizer, index-only scans, multicolumn statistics. Transactions, Nested Transactions (via savepoints). Multi-Version concurrency Control (MVCC). Parallelization of read queries and building B-tree indexes. Table partitioning. All transaction isolation levels defined in the SQL standard, including Serializable. Just-in-time (JIT) compilation of expressions.
- Reliability, Disaster Recovery: Write-ahead Logging (WAL). Replication with Asynchronous, Synchronous, Logical. Point-in-time-recovery (PITR), active standbys. Tablespaces.
- Security: Authentication with GSSAPI, SSPI, LDAP, SCRAM-SHA-256, Certificate, and more. Robust access-control system. Column and row-level security. Multi-factor authentication with certificates and an additional method.
- Extensibility: Stored functions and procedures. Procedural Languages: PL/pgSQL, Perl, Python, and Tcl. There are other languages available through extensions, e.g. Java, JavaScript (V8), R, Lua, and Rust. SQL/JSON path expressions. Foreign data wrappers: connect to other databases or streams with a standard SQL interface. Customizable storage interface for tables. Many extensions that provide additional functionality, including PostGIS.
- Internationalisation, Text Search: Support for international character sets, e.g. through ICU collations. Case-insensitive and accent-insensitive collations. Full-text search.

This database system will manage the data regarding the users, the scenarios configured by the users, and the data from the simulations.

- **REDIS**

Redis is an open source (BSD licensed), in-memory data structure store used as a database, cache, message broker, and streaming engine. Redis provides data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes, and streams. Redis has built-in replication, Lua scripting, LRU eviction, transactions, and different levels of on-disk persistence, and provides high availability via Redis Sentinel and automatic partitioning with Redis Cluster.

You can run atomic operations on these types, like appending to a string; incrementing the value in a hash; pushing an element to a list; computing set intersection, union and difference; or getting the member with highest ranking in a sorted set.

To achieve top performance, Redis works with an in-memory dataset. Depending on your use case, Redis can persist your data either by periodically dumping the dataset to disk or by appending each command to a disk-based log. You can also disable persistence if you just need a feature-rich, networked, in-memory cache. Redis supports asynchronous replication, with fast non-blocking synchronization and auto-reconnection with partial resynchronization on net split. REDIS has been utilized inside the Vensim server in order to store and manage the simulation data coming from the Vensim launches of the model.

- **Celery**

Celery is a task queue with focus on real-time processing, while also supporting task scheduling. Task queues are used as a mechanism to distribute work across threads or machines. A task queue's input is a unit of work called a task. Dedicated worker processes constantly monitor task queues for new work to perform. Celery communicates via messages, usually using a broker to mediate between clients and workers. To initiate a task the client adds a message to the queue, the broker then delivers that message to a worker. A Celery system can consist of multiple workers and brokers, giving way to high availability and horizontal scaling. [it] is written in Python, but the protocol can be implemented in any language [(current clients in NodeJS, PHP)]. In other words, the entities involved in Celery are:

- Producers: also called clients, they are the ones requesting tasks and doing something with the results.
- Broker: the broker is the message transport, used to send and receive messages between producers and workers. In other words, they store the task queue. Celery supports a myriad of message brokers, but currently only two are feature-complete: Redis and RabbitMQ.
- Workers: the workers are processes that constantly watch the task queue and execute tasks.
- Result backend: a backend is only necessary when we want to keep track of the tasks' states or retrieve results from tasks. A result backend is optional but turned on by default.

For our case, Celery will run the simulations in Vensim, and manage the results of the simulations, storing them with the REDIS system, all together.

- **JavaScript and React**

React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces based on components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. React can be used to develop single-page, mobile, or server-rendered applications with frameworks like Next.js. Because React is only concerned with the user interface and rendering components to the DOM, React applications often rely on libraries for routing and other client-side functionality. React contains the following features:

- Declarative: React adheres to the declarative programming paradigm. Developers design views for each state of an application, and React updates and renders components when data changes. This is in contrast with imperative programming.
- Components: React code is made of entities called components. These components are modular and reusable. React applications typically consist of many layers of components. The components

are rendered to a root element in the DOM using the React DOM library. When rendering a component, values are passed between components through props (short for "properties"). Values internal to a component are called its state. The two primary ways of declaring components in React are through function components and class components.

- Function components: Function components are declared with a function (using JavaScript function syntax or an arrow function expression) that accepts a single "props" argument and returns JSX. From React v16.8 onwards, function components can use state with the useState Hook.
- React Hooks: Hooks are functions that let developers "hook into" React state and lifecycle features from function components. Notably, Hooks do not work inside classes, they let developers use more features of React without classes.
- Class components: Class components are declared using ES6 classes. They behave the same way that function components do, but instead of using Hooks to manage state and lifecycle events, they use the lifecycle methods on the React.Component base class.
- Virtual DOM: Another notable feature is the use of a virtual Document Object Model, or Virtual DOM. React creates an in-memory data-structure cache, computes the resulting differences, and then updates the browser's displayed DOM efficiently. This process is called reconciliation. This allows the programmer to write code as if the entire page is rendered on each change, while React only renders the components that actually change. This selective rendering provides a major performance boost.

5.5. ARCHITECTURE

The Model Explorer tool has been implemented following a client-server architecture. Figure 109 shows the general software architecture of the tool which could be represented by the three main typical layers: (i) Data layer; (ii) Business Logic Layer; and (iii) Application Layer. The first two layers of the tool architecture, comprise the tool backend, while the Application Layer represents the tool frontend or Graphical User Interface (GUI) making possible the user interaction.

The server with the tool communicates with the clients using HTTP REST protocol. User access with a user account is managed using Keycloack as data manager given user the availability to save developed policy action scenario. The system is connected with a PostgreSQL database of pre-simulated results of WILIAM model and also with a server for real time simulations using a Multicontext DLL Vensim license.

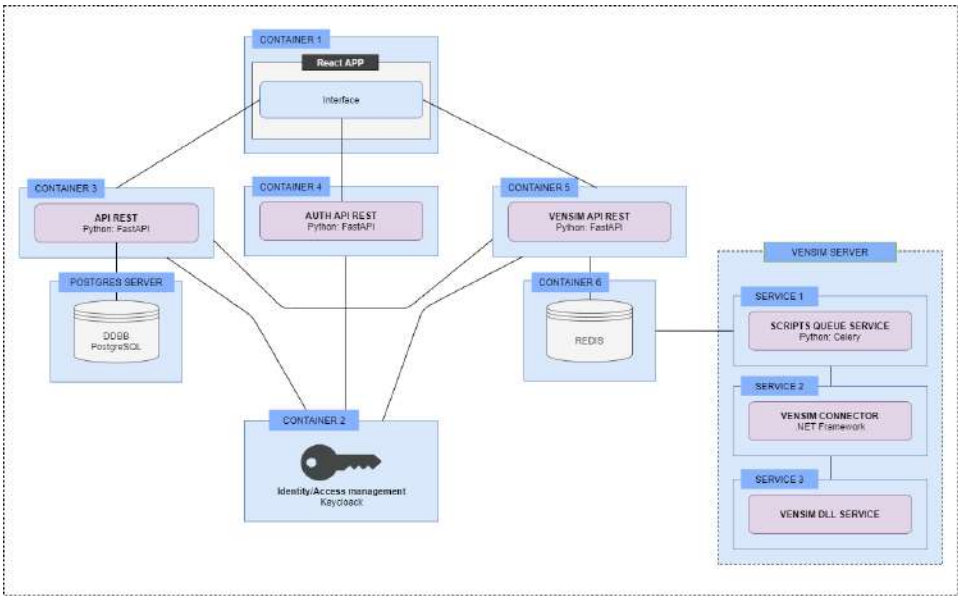


Figure 109: General architecture of Model Explorer tool.

For the backend, inside CARTIF facilities there are a set of servers running in order to perform the related activities of the tool. There is one Windows server (named as VENSIM SERVER in the previous figure) running the application that works as frontend, and also the code that connects to the API. Another server works as a PostgreSQL storage for multiple databases, and concretely the one with the values of the indicators obtained with the simulations obtained with the given parameters from the policies.

In parallel, a Windows server has been deployed in order to generate the simulation data necessary to cover all the possible cases given by the combination of policies. The reason is that the DSS version of Vensim can only run with compiled .dll versions of the model in Windows environments.

The application/interface is developed using the REACT framework to create a user-friendly view for policy implementation and output visualization. Through the API, it can gather the necessary data from the database in order to show the values of the indicators or tool output for each given set of policies that configure the policy action scenario. Post-request action is available to save and connect the policy action scenarios created by the user thanks to the tool for a later use.

5.6. TOOL IMPLEMENTATION

The tool implementation was a collaboration between different partners and subcontracted companies making feasible the tool implementation.

EBAVS/ worked in close collaboration with Ondeuev to build the fronted of the tool, with supervision from CARTIF and CREAF to integrate potential improvements refining the initial layout. In addition, CARTIF worked in the implementation of the functionalities of the frontend components developed by EBAVS/ in REACT framework. The tool backend was developed by CARTIF integrating to different approaches: the first one, storing and managing pre-simulated results from WILIAM model and the second one, running simulations in real time with a Vensim server license. The pre-simulated results from WILIAM model was a joint effort of simulation developed by CARTIF and UVa to collect results using the combination of a set of 12 policies selected by WILIAM modellers.

Finally, the tool was deployed and tested in an internal server detecting the bugs and potential improvements to deliver a final product covering the requirements developed during the initial steps of the tool definition.

5.7. QUALITY ASSURANCE

Several key aspects have been considered in order to ensure a correct and robust user experience for the frontend of the platform.

Performed Quality Assurance (QA) tests for the frontend of the platform includes:

- Functionality Testing: Ensuring that all interactive elements (buttons, forms, and links) work as expected. This kind of tests are fundamental in order to avoid any malfunctioning or incorrect functionality that it could lead to a frustrating user experience.
- User Interface (UI) Testing: In these tests, the correct style, position, and align of the different UI elements are validated. It is crucial for providing users a visually pleasing and consistent experience.
- Cross-Browser Compatibility: The platform has been tested on different web browsers, in order to guarantee that the application functions consistently across a wide range of user environments.
- Security Testing: Identifying and addressing security vulnerabilities has been done, in order to prevent potential breaches.

- Integration with Backend: Proper communication between frontend and backend components has been validated in order to ensure that data is accurately retrieved and submitted, contributing to a seamless user experience.

On the other hand, integration and unit tests play a crucial role in ensuring the reliability and functionality of backend systems.

Integration tests focus on assessing the interactions between different components, modules, or services within the backend, verifying that they work harmoniously together as intended. On the other hand, unit tests target individual units of code, examining their behaviour in isolation to guarantee their correctness.

Employing both integration and unit tests, have been simplified early issues identification by developers, in the development cycle, prevent bugs from propagating into production, and promote code stability. Furthermore, integrating these tests into a Continuous Integration (CI) workflow through platforms like GitLab (in our case) enhances the development process. This integration automates the testing process, ensuring that tests are run automatically whenever new code is pushed to the repository. This not only reduces the risk of introducing defects but also has been provided prompt feedback to developers, facilitating rapid iteration and the delivery of high-quality software.

Both integration and unit tests are integrated in CI in the platform repository.

Keycloak has been used as authentication and authorization server in order to improve QA. External authentication server, like Keycloak, enhance the security, user experience, and overall quality of the global application. It allows to focus on the application's core features during QA testing while relying on a well-established and feature-rich authentication solution.

Some of the Keycloak main features are:

- Security: Keycloak provides robust security features, including authentication and authorization. By using Keycloak, -standard security protocols such as OAuth 2.0 and OpenID Connect can be implemented, ensuring that user identities are managed securely.
- Centralized User Management: Keycloak allows to manage user identities, roles, and permissions in a centralized manner. This simplifies user management and reduces the risk of inconsistencies in access control.
- Single Sign-On (SSO): Keycloak supports SSO, allowing users to authenticate once and then access multiple applications without needing to log in again. This improves user experience and reduces the need for users to remember multiple sets of credentials.
- Federated Identity: Keycloak supports federated identity, enabling users to log in using their existing accounts from external identity providers like Google, Facebook, or others. This can improve user convenience and reduce the need for users to create new accounts.
- Token-Based Authentication: Keycloak issues JSON Web Tokens (JWTs) upon successful authentication. These tokens can carry user claims and access permissions. By validating these tokens in the application, ensure secure access to different parts of your application (especially in the backend) based on user roles and permissions.
- API Security: Using Keycloak can help secure APIs by ensuring that only authenticated and authorized users can access them.
- Testing Efficiency: Having a well-established authentication server like Keycloak allows to focus the testing efforts on the application's core functionality rather than spending time on building and testing authentication mechanisms.

- Scalability and Maintenance: By offloading authentication and user management to Keycloak, a separation of concerns and improve the scalability of the application can be achieved. Maintenance of user data and security updates can be managed separately from your application codebase.

5.8. DEVELOPER GUIDE AND DEVELOPMENT

5.8.1. FRONTEND DEVELOPMENT GUIDE

For develop in frontend code the next dependencies should be installed:

- Node:16.14
- npm 8.3.1
- React: 17.0.2

5.8.2. BACKEND DEVELOPMENT GUIDE

5.8.2.1. REQUIREMENTS

For develop in backend code the next dependencies should be installed:

- Python 3.8.10
- Poetry >= 1.2.2

5.8.2.2. DATABASE

A database (Figure 110) is needed in order to work with the pre-simulated series. The structure of the Database is shown in the next diagram:

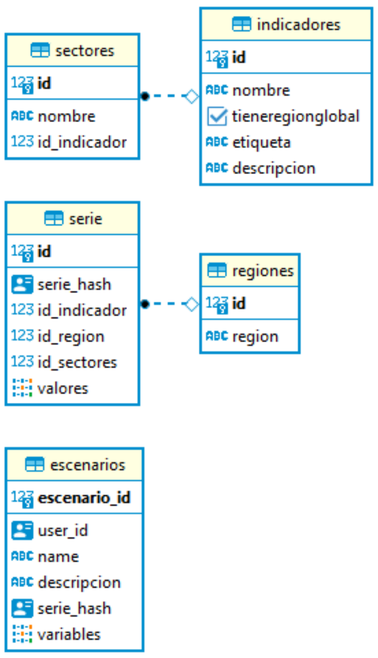


Figure 110: Architecture of Model Explorer tool database.

5.8.2.3. AUTHORIZATION/AUTHENTICATION SYSTEM

A Keycloak deployment is needed in order to work with authentication/authorization. The image user for this project is jboss/keycloak:16.1.1.

A realm creation is needed and a couple of clients (locomotion-frontend and locomotion-backend) with public and confidential configuration respectively.

5.8.2.4. ENVIRONMENT VARIABLES

The next Environment Variables should be included in an .env file in order to develop in backend code:

- API_PORT: External port for Main API (only needed in deploy)
- AUTH_API_PORT: External port for Auth API (only needed in deploy)
- CELERY_BOKER_URL: redis://redis_server:port/0
- CELERY_RESULT_BACKEND: redis://redis_server:port/0
- CORS_ORIGINS: Frontend public URL. I could be * in development environments
- DB_DATABASE: Database name
- DB_HOST: Database host
- DB_PASSWORD: Database password
- DB_USER: Database user
- EXTERNAL_ROOT_PATH_API: Subroute path in case of publish in a subroute URL ("/" in development enviroments or if it is published in root domain URL)
- EXTERNAL_ROOT_PATH_AUTH_API: Same than above for Auth API
- EXTERNAL_ROOT_PATH_VENSIM_CELERY_API: Same than above for Vensim API
- KEYCLOAK_ANONYMOUS_PASSWORD: Anonymous user password
- KEYCLOAK_ANONYMOUS_USERNAME: Anonymous user username
- KEYCLOAK_BASE_URL: Keycloak base url
- KEYCLOAK_CLIENT_ID_BACK: Keycloak backend client id
- KEYCLOAK_REALM_ID: Keycloak realm name
- KEYCLOAK_SECRET_KEY_BACK: Keycloak realm secret
- REDIS_PORT: Redis port
- VENSIM_API_PORT: Vensim API port

After clone the project, the next command should be executed for each API of the de backend ("api", "auth_api" and "vensim_api"), in order to prepare the development python environment:

```
poetry install
```

All the APIs includes online documentation (Swagger / OpenAPI 3) in order to simplify the development and frontend integration process.

The main "auth_api" endpoints are included in this document in order to simplify the initial user setup:

5.8.2.5. REGISTER A NEW USER

```
curl --location '{{base_url}}/auth/user' \
--header 'Content-Type: application/json' \
--data-raw '{
  "username": "username",
  "email": "email",
  "credentials": [
    {
      "value": "userpassword",
      "type": "password"
    }
    ],
    "enabled": true
}'
```

Enabled is used to activate the account immediately or manual enable the account by a Keycloak administrator.

5.8.2.6. LOGIN

```
curl --location '{{base_url}}/auth/login' \
--header 'Content-Type: application/json' \
--data '{
    "username": "username",
    "password": "userpassword"
}'
```

5.8.2.7. FORGOT PASSWORD

```
curl --location --request PUT '{{base_url}}/auth/reset-password' \
--header 'Content-Type: application/json' \
--data-raw '{
    "email": "rafdel@cartif.es"
}
```

5.8.2.8. UPDATE USER

```
curl --location --request PUT '{{base_url}}/auth/user' \
--header 'Content-Type: application/json' \
--header 'Authorization: token' \
--data-raw '{
    "username": "username",
    "email": "username@mail.com"
}
```

5.8.2.9. REFRESH TOKEN

```
curl --location 'teide.cartif.local:8014/auth/refresh-token' \
--header 'Content-Type: application/json' \
--header 'Authorization: current_token' \
--data '{
    "refresh_token": refresh_token
}
```

5.9. DEPLOYMENT AND OPERATION INSTRUCTIONS

Continuous Integration (CI) workflow has been implemented in order to simplify development and deployment process in a dockerized environment.

In order to deploy the frontend and backend applications without using CI, a text file should be filled with the environmental variables needed in the deploy process (see ENVIROMENTAL VARIABLES section).

The next command prepares the docker containers needed and deploy them:

```
docker-compose --env-file env_vars.txt up -d --build
```

6. SYNERGIES AND POTENTIAL IMPROVEMENTS

6.1. SYNERGIES ACROSS LOCOMOTION TOOLS DEVELOPMENT

The design of the three applications was agreed during the early stages of development, so that the three LOCOMOTION tools would not only share the branding and general design, but also a similar look and feel. Ondeuev was in charge of creating such homogeneous design, while adapting it to the particular needs of each app. All developers of the three apps met regularly with Ondeuev to guarantee the consistency in their designs.

The result has made it possible to give a project image to all the applications using similar elements, colors and layout between the applications so that the user can switch from one to another, reducing the learning curve. This can be clearly seen by comparing the designs included in this document.

Another relevant point to consider has been the use of similar approaches in the use of simplified scenarios for the Model analyzer and the Model Explorer. This simplified scenario together with the implementation of the connection of the model through a license installed on a server, allows to increase the degrees of freedom of the user in the use of the application in the case of the Model Explorer and to reduce the complexity of use in the case of the Model

Analyzer. Finally, we must highlight the use of a pre-calculated database of model simulations using a simplified scenario of 12 policies and all their combinations. This database is the basic engine for both the Model Explorer and the Crossroads II Game: For this reason, great efforts have been invested to make it fully functional with the shortest access times. This database, designed and created by CARTIF, will be exposed through an API for use in both applications.

6.2. FUTURE WORK

Although all developments for the **Model Analyzer** have long been finalized, there are still some tasks that will need to be addressed until the end of the project. Such pending tasks arise from the delays experienced in the release of a stable version of the underlying WILIAM model and the corresponding scenarios. To adapt to this inconvenience, the development of the app was made using test models (including early releases of WILIAM) and made-up scenario parameters and output variables.

The remaining tasks until the end of the project include:

- Deploy the model Analyzer with latest version of WILIAM (v1.1) and generate the simplified and full scenarios in the Data Client.
- Testing that higher order matrices and time-series are displayed correctly, and that changes to their values are correctly considered.
- Testing the application with the latest version of WILIAM and with higher loads (more concurrent users and larger payloads).
- Refining the list of scenario parameters and output variables to further improve the usability of the application and the relevance of the outputs obtained with it.
- Setting stricter upper and lower bounds for scenario parameters, to ensure model stability.
- Improving metadata associated to scenario parameters and output variables (variable names, units, descriptions, categories and subcategories, dimension names, etc). This will progressively improve the user experience in further releases.
- Creating and uploading the application executables (for Windows, Mac and GNU-Linux) in the project website, and adding links to the original source code of the app.

- Making the repositories where the code is maintained public.

The **Model Explorer** is an application that is still under development due to delays in the implementation of the WILIAM model. The version of the application deployed as of the date of this deliverable is based on the version of the WILIAM 1.0 model, having 11,000 simulations with variations in 12 policies and more than 20 indicators or model outputs on which to evaluate the effect of the implementation of policies. The continuous delays in the model have created a bottleneck for the development of the application since it is necessary to have a stable version of the model on which to simulate for storing the results in the application database. Among the pending activities for the completion of the application, the following should be highlighted:

- Carry out simulations with version 1.1 of the WILIAM model in order to adapt the results to the use of 20 policies. The infrastructure for the simulation is available and it is expected that sufficient final simulations will be available before the completion of the model.
- Integration of the real-time model with a Vensim Munticontext DLL license to allow simulations with multiple users. In this way, it will be guaranteed to obtain results in case the number of simulations does not cover all the variations between policies. Process management to guarantee that both approaches can be executed simultaneously considering the availability or not of simulations.
- Adjustment in the display of the outputs to offer an optimal user experience.
- Deploy the tool in the project website and adding links to the original source code of the app.
- Making the repositories where the code is maintained public.

The **Global sustainability Crossroads II Game** has a functional version based on the simulations developed with the MEDEAS model, which is the predecessor of the WILIAM model. Having this database of simulations has made it possible to solve the delays in the implementation of the model. However, these simulations will need to be replaced by the ones being configured during the Model Analyzer development, before the end of the project. In this way, the pre-simulated results database will be a database of model results common for both applications, which allows saving efforts and capitalizing on developments between both applications. As in the other two tools, the game will be deployed in the project website including links to the original source code of the app.

7. ACCESS TO LOCOMOTION TOOLS

All the tools of LOCOMOTION project will be available through the website and more specifically through the online space provided on it for each specific tool:

- Model Analyser: https://www.locomotion-h2020.eu/locomotion-models/model-analyser/
- Global Crossroads II Game: https://www.locomotion-h2020.eu/locomotion-models/global-sustainability-crossroads-ii/
- Model Explorer: https://www.locomotion-h2020.eu/locomotion-models/model-explorer/

At the time of writing this document, the first release of the tools is available being updated in the remaining three months until the end of the project including some specific request collected from the testing and validation activities with the stakeholders.

CONCLUSIONS

This document, integrates the results of Task 11.1 (Decision-making, gamification and awareness tools) where the design, development and deployment of LOCOMOTION tools were developed.

Throughout the document, a detailed analysis of the methodology implemented for the development of the tools is carried out, considering all the necessary stages for the implementation of the code that has finally made it possible to have three tools that complement and add value to the WILIAM model. Each tool is focused on a target audience and therefore a completely different user iteration has been designed according to the target group, clearly conditioning the developments and complexity in the use of the model and guiding its integration in the tool.

In addition to the development's description, a next steps section has been included where the pending aspects in the development of the tools are raised and that will be updated in the remaining three months until the end of the project. The most critical aspects are related to updating the model results. The delay in its implementation has generated a very important bottleneck for the implementation of the tools with an accumulated delay of about a year. It must be considered that the functional version of the model was released in July 2023, so, until then, it had to work with partial versions which changed very substantially with respect to the version 1.1 of the WILIAM model. Being able to overcome these deficiencies associated with the absence of a functional model has been a challenge which the developers of the tools have managed to solve by working on the standardization of the formats for managing the policy inputs and the indicators to evaluate their effects.

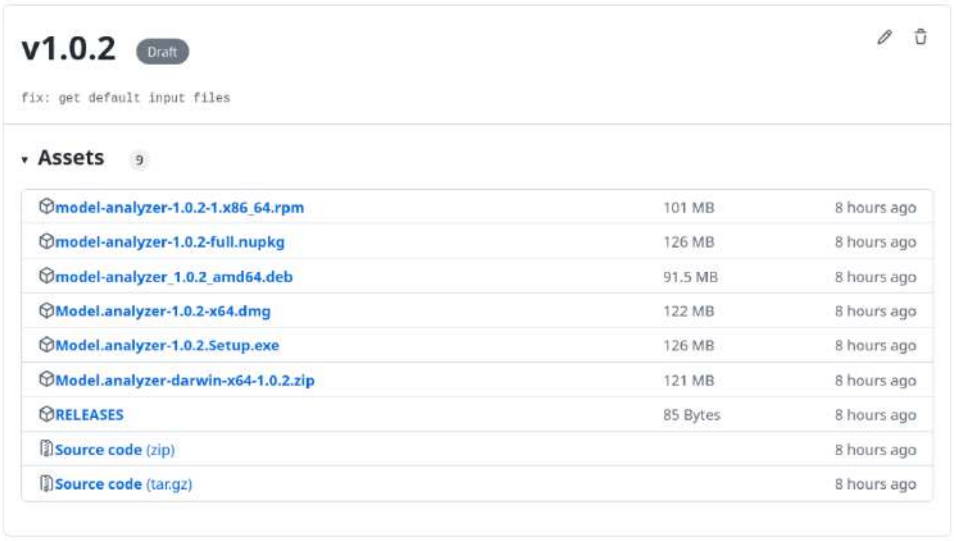
Finally, the document includes in its annexes a small user manual that can help to understand the use of each tool. These manuals are complemented by the help included in each of the tools.

ANEXES: USER MANUALS

MODEL ANALYSER

INSTALLATION INSTRUCTIONS

The Model Analyzer Desktop app is installed from the executables that are available from the project website or from the Releases section in the Github repository (https://github.com/locomotionH2020/model_analyzer/releases). Executables are available for GNU/Linux (deb and rpm), Windows (exe) and Mac (dmg). To see the installers for all platforms, click on Assets, right below the name of the latest release (Figure 111).



v1.0.2 Draft

fix: get default input files

Assets 9

model-analyzer-1.0.2-1.x86_64.rpm	101 MB	8 hours ago
model-analyzer-1.0.2-full.nupkg	126 MB	8 hours ago
model-analyzer_1.0.2_amd64.deb	91.5 MB	8 hours ago
Model.analyzer-1.0.2-x64.dmg	122 MB	8 hours ago
Model.analyzer-1.0.2.Setup.exe	126 MB	8 hours ago
Model.analyzer-darwin-x64-1.0.2.zip	121 MB	8 hours ago
RELEASES	85 Bytes	8 hours ago
Source code (zip)		8 hours ago
Source code (tar.gz)		8 hours ago

Figure 111: Installers for all platforms.

USER SIGN-UP AND SIGN-IN

To be able to use the Model Analyzer, the user must create an account. To do so, click on the SIGNIN button (Figure 112).

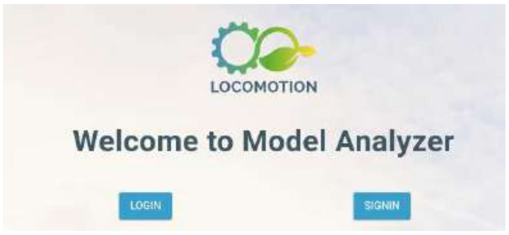


Figure 112: Login and Signin buttons.

In the registration page (Figure 113), enter you email and a password, and click on the SIGNIN button.

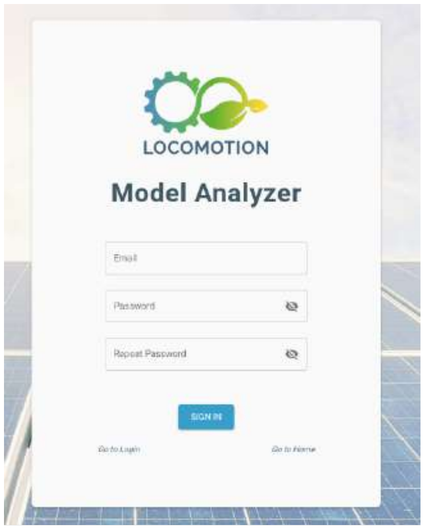


Figure 113: Registration page.

You will receive an email to the email address you provided, asking you to confirm the registration.

Beware that the confirmation email may be tagged as spam by your email service provider. If you do not receive the confirmation email after a few minutes, please contact the system administrator.

Once you have confirmed your registration, Log-in to the Model Analyzer app using your email and password (Figure 114).

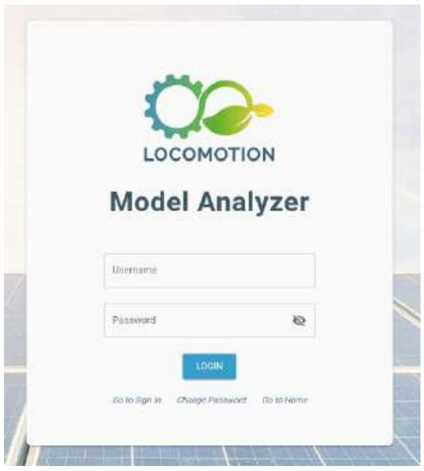


Figure 114: Login page.

Once logged in, the session will only expire after 24 hours, after which you will need to log back in. To logout manually, click on your email at the top right corner of the screen and select Logout.

DELETING A USER ACCOUNT

To delete your user account and all data (i.e., scenarios and simulation results) associated with it, click on your email account at the top right corner of the app, and select *Delete User*.

APPLICATION MENUS

The following instructions can be consulted from within the Model Analyzer app, by clicking on the question mark at the top right corner of the interface, once logged in.

- SCENARIOS

From this menu you can visualize the scenarios that are available in your system, their current status, and the values of their parameters. If it is the first time you open the app, you will only have the Full and Simplified default scenarios.

As the name implies, the Full scenario allows for further customization than the Simplified scenario. The main difference between the two is that in the Simplified scenario, the dimensions for most parameters have been aggregated to ease configuration. Other minor differences exist, mostly in terms of the parameters you can modify, but that I will let you discover by yourself.

Once you have created and run new scenarios, they will be displayed using different background colors:

- White: the scenario was created but it has not yet been simulated (results not available).
- Yellow: the scenario has been simulated, but the results are not yet available in your system. Go to Manage Scenarios and see the status of your scenario in "In the Cloud" table)
- Green: the simulation succeeded and the results were downloaded to your system.
- Red: the simulation failed. Please launch the simulation again in a few minutes, and if the issue persists, contact the system administrator.

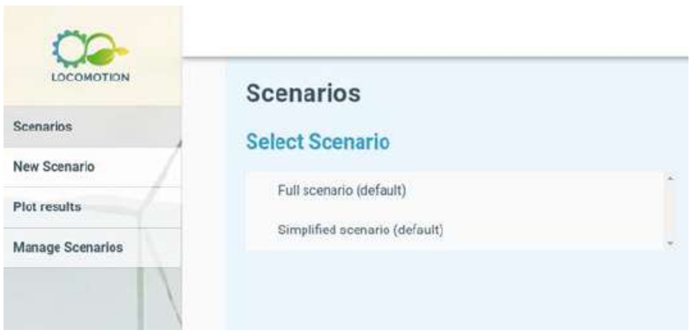


Figure 115: Scenarios list.

To visualize the parameters values of a particular scenario, click on it on the list.

Then start moving along the different tabs, which correspond to the different WILIAM modules.

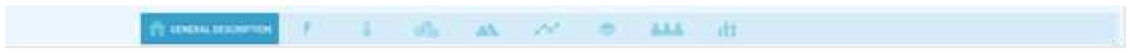


Figure 116: Top scenario menu icons.

Note that to see the values of the parameters, you first need to select a region from the dropdown menu on the left.

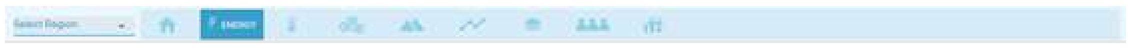
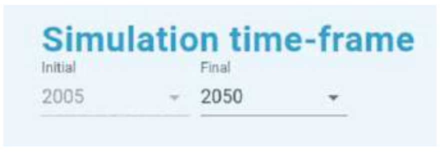


Figure 117: Select region dropdown.

- NEW SCENARIO

From here you will be able to create and parametrize new scenarios.

The first thing you need to do to create a new scenario is to select the final date of the simulation (the initial date is year 2005 and cannot be modified).


Simulation time-frame

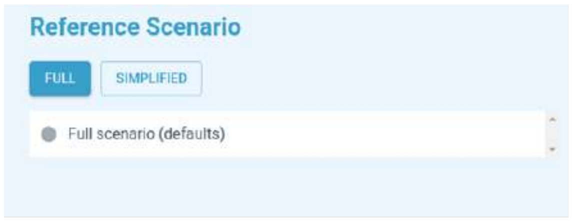
Initial: 2005

Final: 2050

Figure 118: Initial and final simulation times selectors.

Then, select the reference scenario for your new scenario. Note that new scenarios need always be based on a pre-existing one, from which they inherit the default values of all parameters.

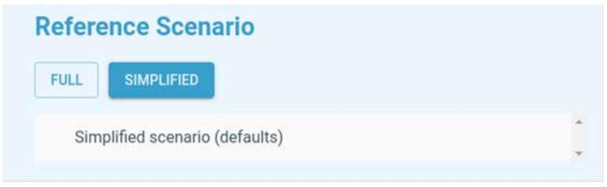
As mentioned in the Scenarios section, when using the tool for the first time, the only available scenarios are Full and Simplified.


Reference Scenario

FULL | SIMPLIFIED

Full scenario (defaults)

Figure 119: Full reference scenario.


Reference Scenario

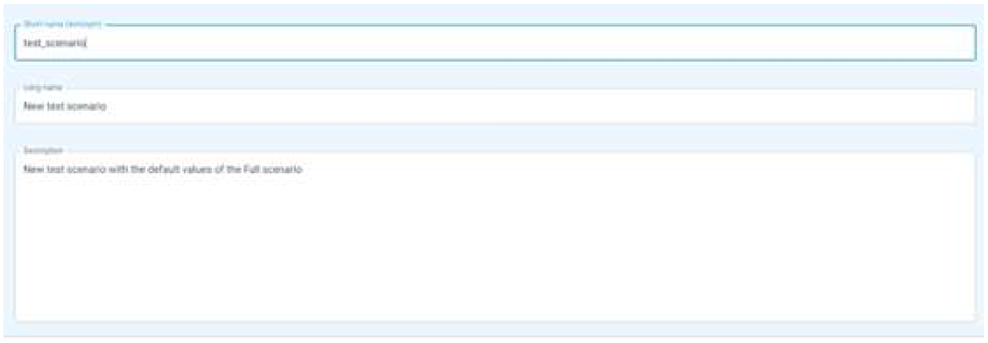
FULL | SIMPLIFIED

Simplified scenario (defaults)

Figure 120: Simplified reference scenario.

After selecting the desired reference scenario, you need to fill in the following information:

- Short name (Acronym): an acronym or very short sentence to describe your scenario (e.g. BaU, EU_GREEN_DEAL, degrowth_scenario). This will be the main identifier of your scenario.
- Long name: this is a longer definition, that may be useful when sharing the scenarios with others, or when obscure Short scenario names are used.
- Description: here you may write what you want to achieve with this scenario, which are its main drivers, etc.


Short name (Acronym): test_scenario

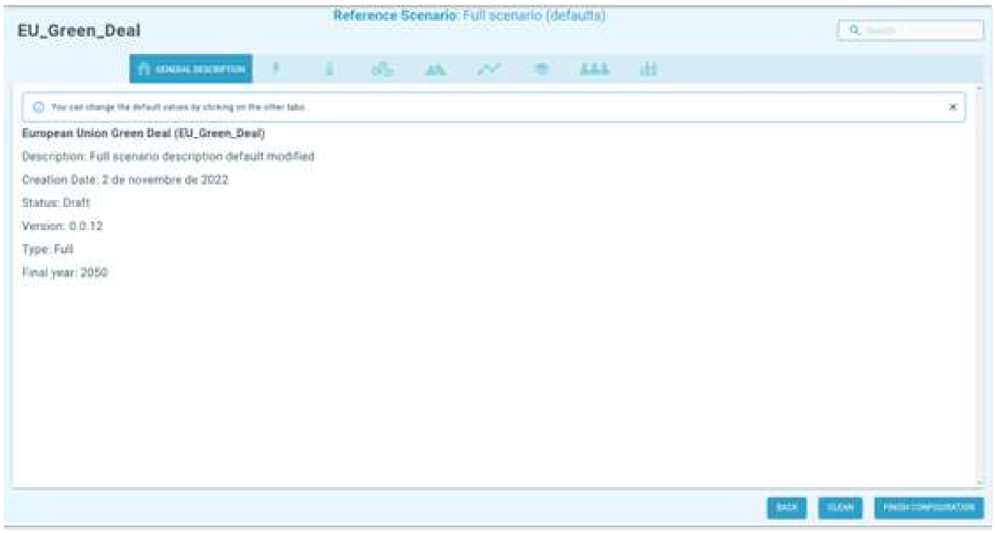
Long name: New test scenario

Description: New test scenario with the default values of the Full scenario

Figure 121: Fields to fill in to create a new scenario.

When all three fields have been filled in, click on Continue to continue with the configuration of your new scenario, or Clear to start over.

If you click on Continue you will land on the summary page for your scenario.


EU_Green_Deal

Reference Scenario: Full scenario (defaults)

You can change the default values by clicking on the other tabs.

European Union Green Deal (EU_Green_Deal)

Description: Full scenario description default modified

Creation Date: 2 de novembre de 2022

Status: Draft

Version: 0.0.12

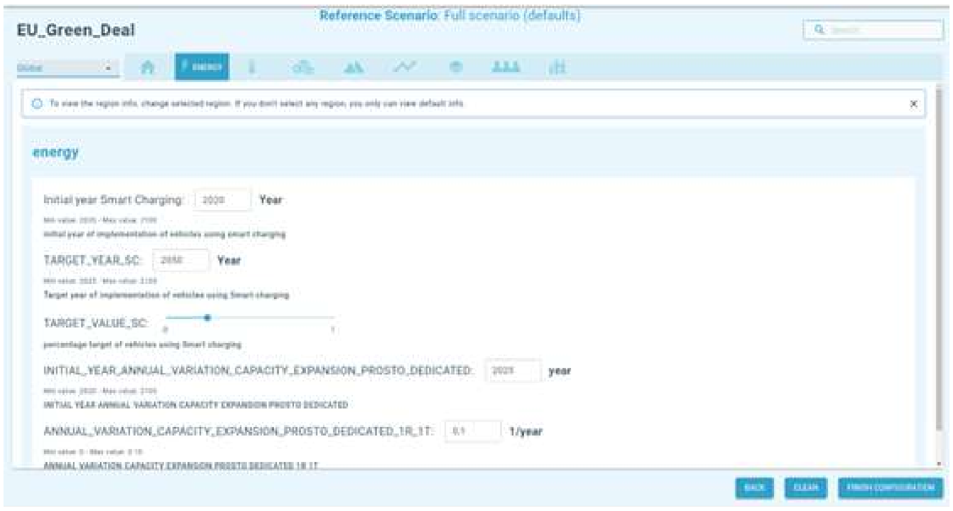
Type: Full

Final year: 2050

Figure 122: Summary description of the draft scenario.

Here, you can see a summary of the metadata you provided for your new scenario. Note that at this stage, the Status is Draft, which indicates that the configuration has not yet concluded.

If you now click on the icons on the top bar, you will be able to configure the scenario parameters for each of the model modules.


EU_Green_Deal

Reference Scenario: Full scenario (defaults)

To view the region info, change selected region. If you don't select any region, you only can view default info.

energy

Initial year Smart Charging: 2020 Year

Min value: 2020 - Max value: 2100

initial year of implementation of vehicles using smart charging

TARGET_YEAR_SC: 2050 Year

Min value: 2025 - Max value: 2100

Target year of implementation of vehicles using Smart charging

TARGET_VALUE_SC:

percentage target of vehicles using Smart charging

INITIAL_YEAR_ANNUAL_VARIATION_CAPACITY_EXPANSION_PROSTO_DEDICATED: 2025 year

Min value: 2020 - Max value: 2100

INITIAL YEAR ANNUAL VARIATION CAPACITY EXPANSION PROSTO DEDICATED

ANNUAL_VARIATION_CAPACITY_EXPANSION_PROSTO_DEDICATED_1R_1T: 0.1 1/year

Min value: 0 - Max value: 0.15

ANNUAL VARIATION CAPACITY EXPANSION PROSTO DEDICATED 1R 1T

Figure 123: Scenario parameters with default values from the reference scenario.

Note, again, that to be able to see the available scenario parameters, you need to select a region from the Select Region dropdown.

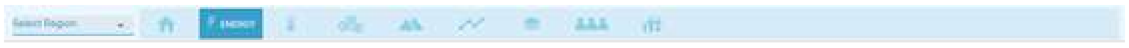
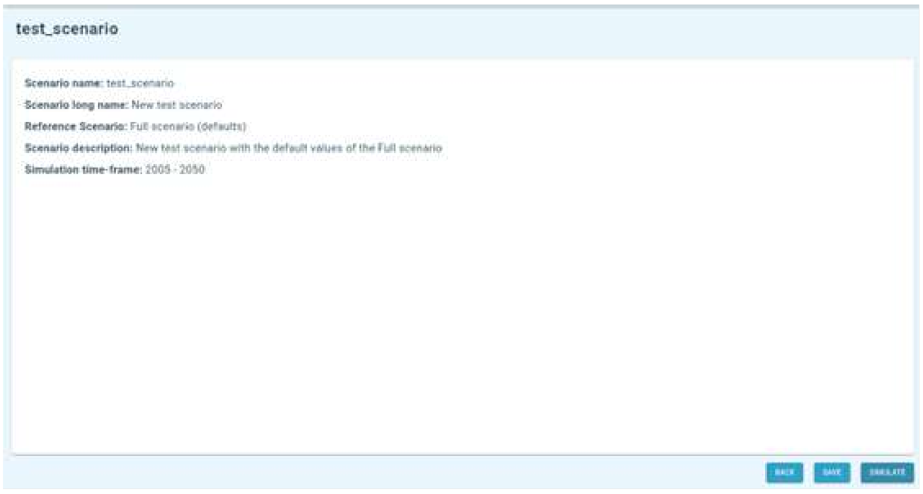


Figure 124: Select region dropdown (left).

When you are done configuring the scenario, click on Finish configuration to move to the final configuration step, Back to go back to the initial scenario configuration page, or Clean to delete all configurations.

By clicking on Finish configuration, you move to the final stage of the scenario creation, where you can either Save the scenario for later use, Simulate the scenario straightaway or go Back to continue editing the scenario.


test_scenario

Scenario name: test_scenario
Scenario long name: New test scenario
Reference Scenario: Full scenario (defaults)
Scenario description: New test scenario with the default values of the Full scenario
Simulation time-frame: 2005 - 2050

Figure 125: Summary of the configuration of the newly created scenario.

If you click on Simulate, the scenario configuration will be automatically saved, and if you move to the Manage Scenarios menu, you can track the simulation's progress.

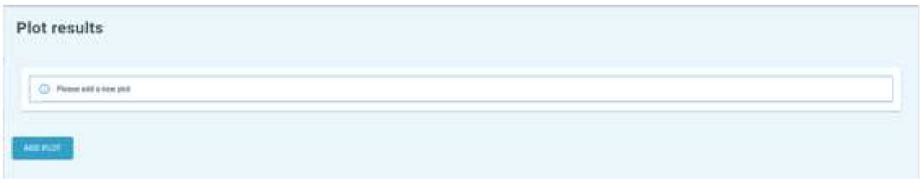
Alternatively, clicking on Save will just save the scenario in your system for later use.

- PLOT RESULTS

From this menu, you can visualize the results of your simulations.

Note that to plot the results associated to a scenario, you must have simulated it beforehand and downloaded the results in your system. For further details, see the Manage Scenarios section.

To add a new plot, click on the Add plot button.


Plot results

Please add a new plot

Figure 126: Adding a new plot.

Then, select the scenario and the output variable you want to visualize. To compare the same output variable for different scenarios, select two or more scenarios from the dropdown.

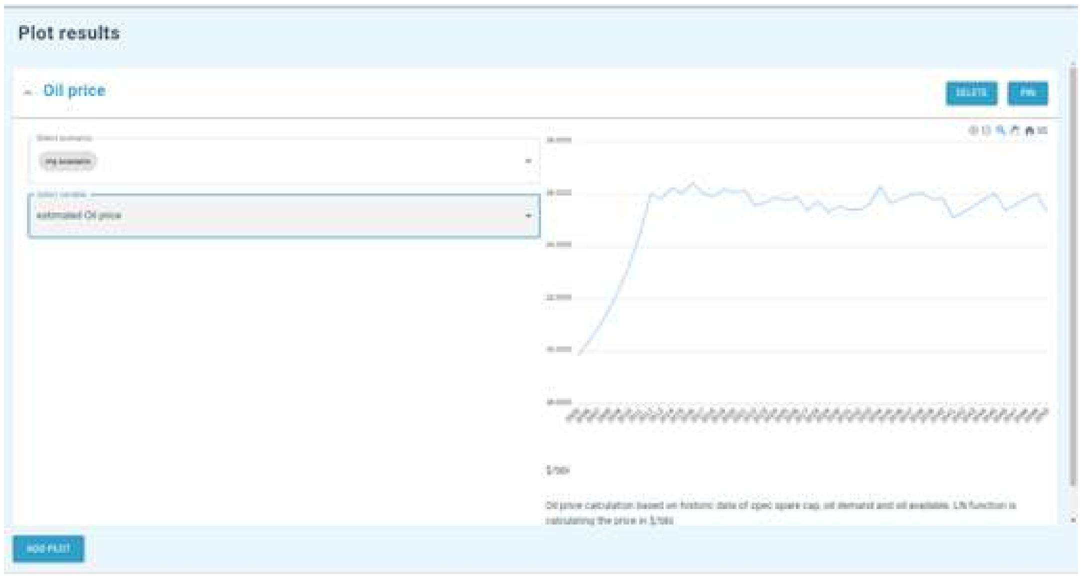


Figure 127: Select the scenario and the output variable you want to visualize.

If the selected output variable does not have any dimensions, it will be displayed on the right panel as in the previous figure. Otherwise, you will be asked to select any or all the dimensions to plot, by placing them to the boxes on the right.

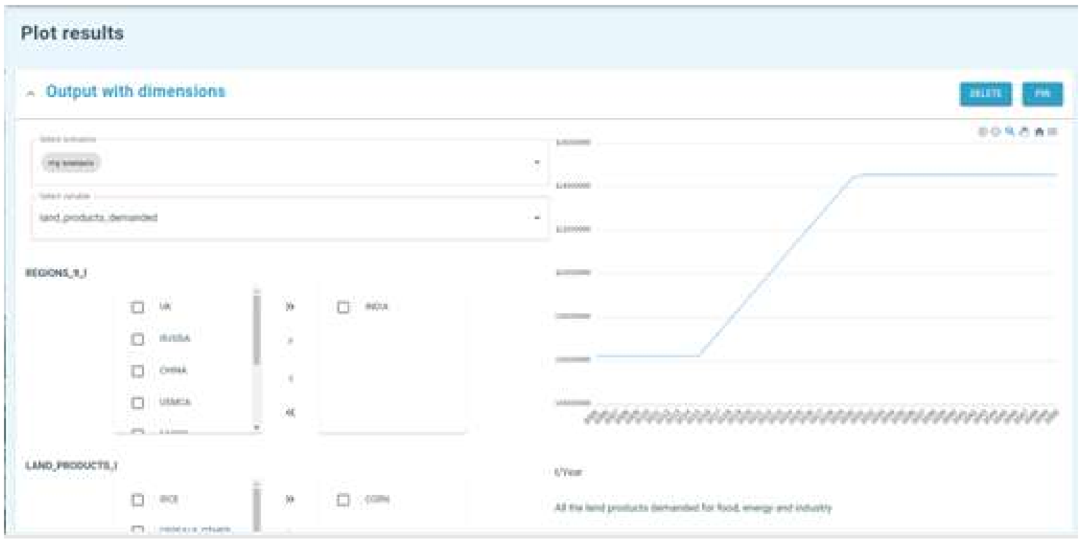


Figure 128: Selecting dimensions to plot.

If you abandon this page, the plots that are not pinned will be erased. To pin a plot for later analysis, click on the Pin button. To delete a pinned plot, click on the Delete button.

The plot tool allows you, among other things, to download the data associated to the plot in csv format and save the figure in png (raster) or svg (vectorial) formats.

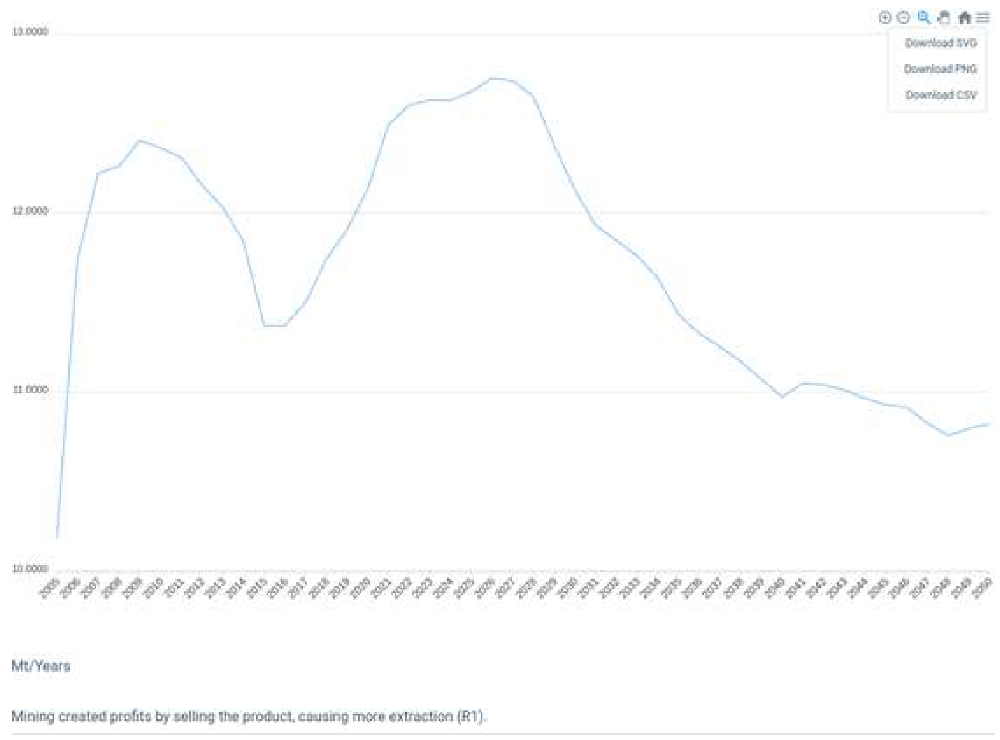


Figure 129: Additional plot options (see top right corner).

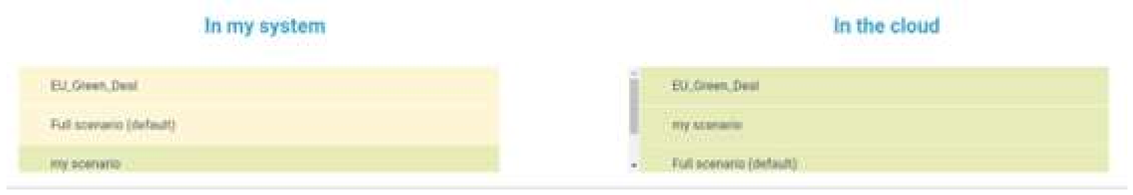
- MANAGE SCENARIOS

In the Model Analyzer, simulations are run on a dedicated server (Cloud from now on), while the least computationally intensive tasks (scenario configuration and results visualization) are performed in the users' systems.

Although an internet connection is required to launch simulations and to download simulation results from the Cloud, the desktop app can still be used offline for creating scenarios and visualizing available simulation results.

Despite the benefits, from a user's perspective this architecture comes at the added cost of having to manage scenarios and simulation results in two distinct locations.

In the Manage Scenarios page, users can manage scenarios and simulation results both locally (in the user's system) and in the Cloud.



In my system	In the cloud
EU_Green_Deal	EU_Green_Deal
Full scenario (default)	my scenario
my scenario	Full scenario (default)

Figure 130: In my system and in the cloud tables.

Scenarios in the user's system are listed in the “In my computer” table, while those in the cloud are listed in the “In the cloud” table. Each scenario is associated with a status, which can be viewed by clicking on the scenario name.

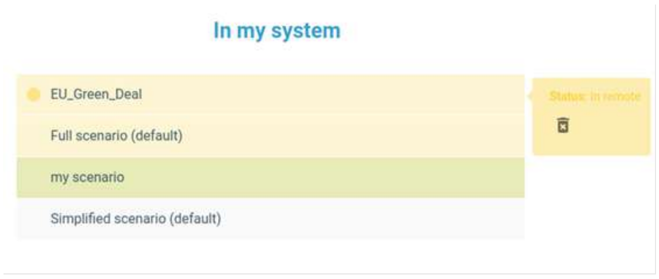


Figure 131: Scenario status.

We will now see the different possible states of the scenarios in each table, their meaning and how they are displayed. We will also see how the scenarios can be managed according to their states.

- SCENARIO STATUS

When a scenario is created (not yet simulated), it resides only on your system, and it is displayed with a white background in the "In my system" table, and with the status Created.

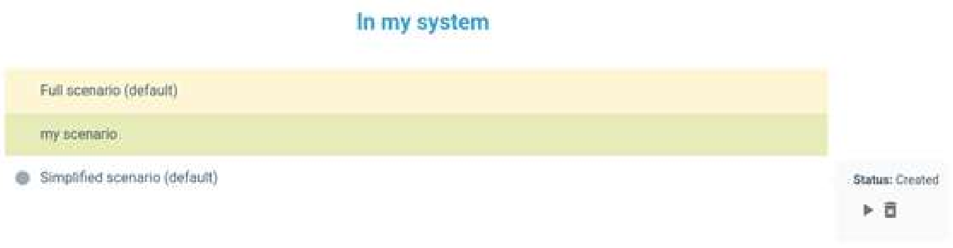


Figure 132: Scenario with status Created.

At this point, the user can simulate the scenario by clicking on the Play button from the menu that opens when clicking on it (see previous image).

After launching the simulation, the state immediately changes to In remote in "In my system" table, and In queue in "In the cloud" table, and the background turns yellow in both tables.

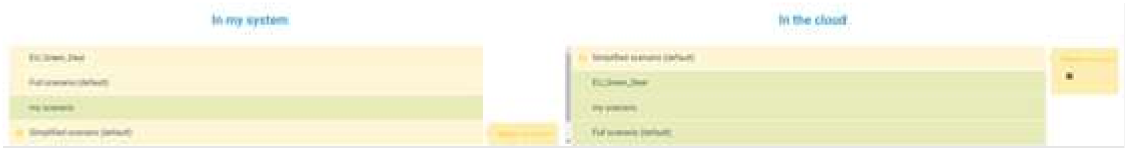


Figure 133: In remote and in queue status.

When the simulation starts running in the cloud, the background colour for the scenario in the "In the cloud" table turns to blue, and the state changes to Running. In the "In my computer" table, the status will still be in remote until the state changes in the cloud.

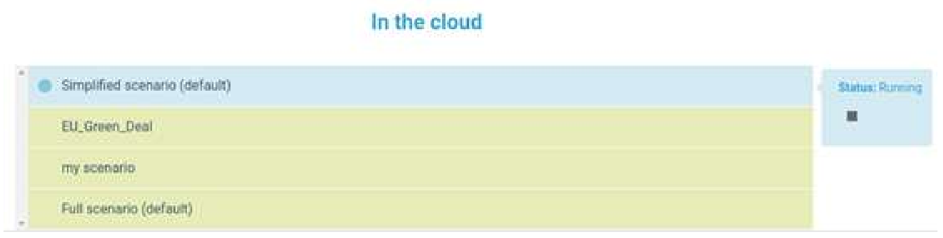


Figure 134: Running simulation.

A Running simulation can be cancelled at any time by clicking on the name of the scenario in the "In the cloud" table and clicking on the Stop button (see previous figure).

If the simulation succeeds, the status changes to Finished and the background colour changes to green only in the "In the cloud" table. This is to signify that the simulation results are available in the Cloud, but they have not yet been downloaded to your machine.

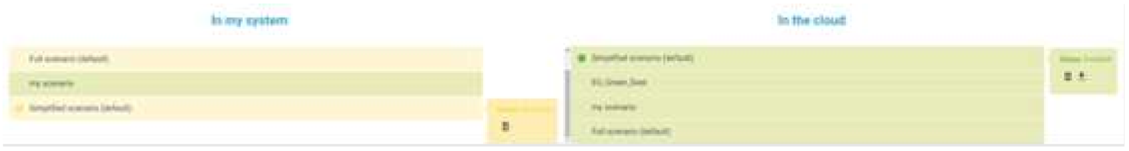


Figure 135: Finished simulation.

At this stage, a download icon becomes available when clicking on the scenario in the "In the cloud" table (see previous image).

When the results are downloaded to your system, the status in the "In my machine" table changes to Available, which means that you can already visualize the results from the Plot results menu.

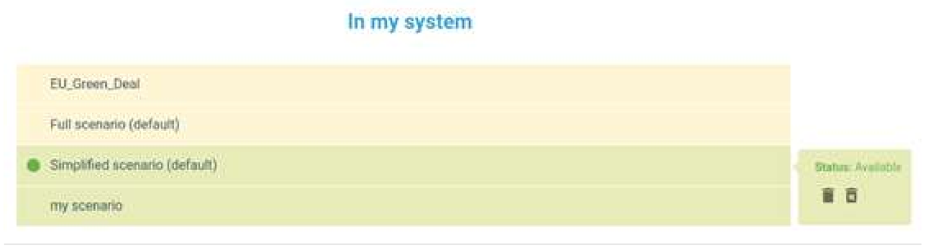


Figure 136: Results Available in the user's system.

On the contrary, if the simulation fails, the background color for the scenario in the two tables changes to red, and the status changes to Failed.

- DELETING SCENARIOS

Your account comes with a limited storage space in the Cloud. That means you can only store a limited number of scenarios and simulation results on it. The percentage of used space can be seen on the top right corner of the itnerface.

Figure 137: Used storage.

If you are close to reaching 100%, you may not be allowed to run any further simulations. In this case, we encourage you to delete old or unused scenarios from the Cloud. This can be achieved by clicking on the name of the scenario you want to delete on the "In the cloud" table, which will open a menu, from which you can delete it by clicking on the bin icon.

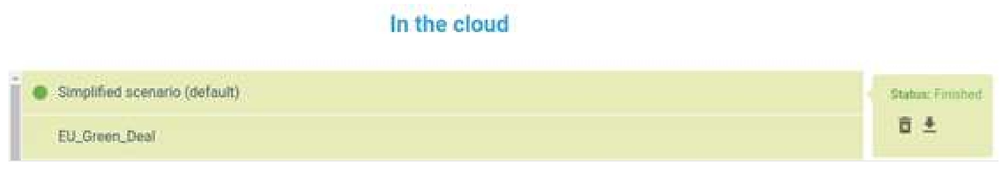


Figure 138: Delete scenario from the Cloud.

Note that this action will delete both the scenario and the simulation results from the Cloud. Hence, remember to download to your system any scenarios you want to keep, before deleting them from the Cloud.

Scenarios can also be deleted from your system. In this case you have two options:

1- Deleting only the scenario results (if they had been previously downloaded) while keeping the scenario configurations (left bin icon).

2- Deleting the scenario and the results (right bin icon).

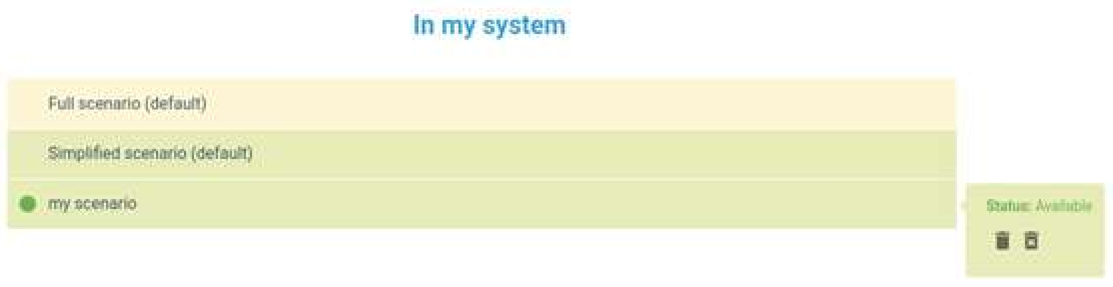


Figure 139: Delete scenario from your system.

CROSSROADS II GAME

The website for accessing the game is http://crossroads.geeds.eu/ . The image in Figure 140 shows the main screen containing three links that allow navigation to three different pages. The first of these navigates to the screen for joining players to a game room (Player's Handbook section), the second of these navigates to the moderator login screen or room creation screen and the third of these navigates to the moderator registration screen (Moderator's Handbook section).

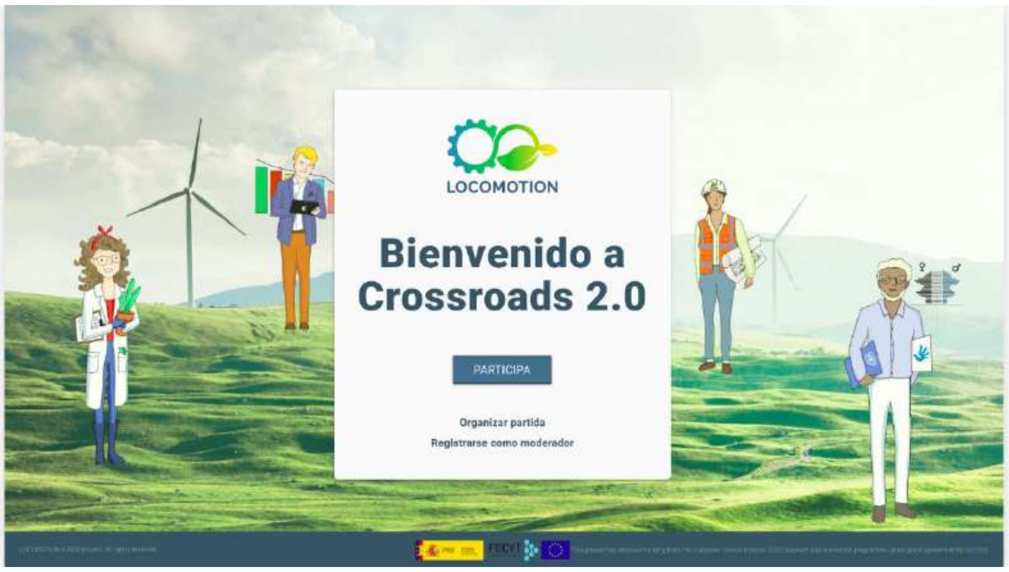


Figure 140: Main screen of Global Crossroads II Game.

Anyone can play the role of moderator. The moderator creates and manages game rooms, i.e. games. The moderator sets the number of teams and the maximum number of people per team. He can also end the games. When he creates a game he gets a room code which is the code that the players need to participate in a game.

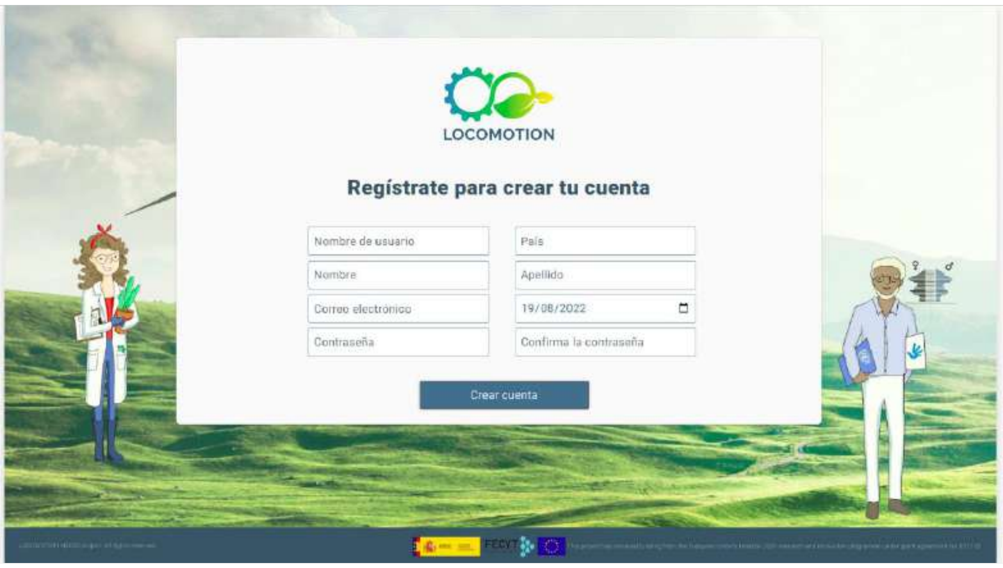


Figure 141: Registration screen.

- MODERATOR MANUAL

If it is the first time that the user accesses the game and wants to create games you have to register. In the registration screen the user has to enter his data in the form in order to create the account (Figure 141). By placing the cursor over each of the form fields, a message appears with the format of the data to be entered in that field. At the bottom of the page, there is a button to submit the form and a link to access the login screen (Figure 142). If the result of the registration is correct, the session is automatically started and the menu page is displayed.

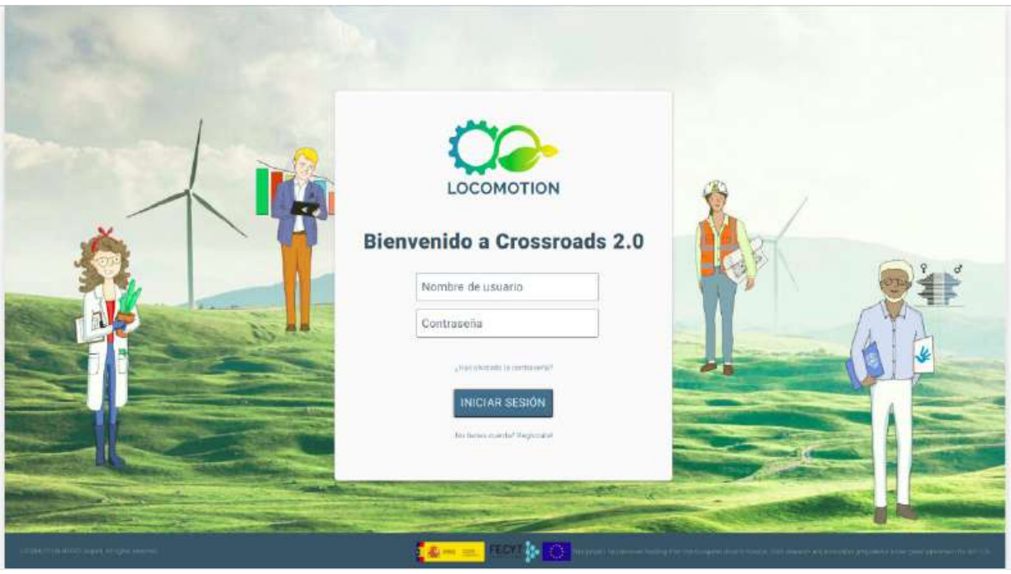


Figure 142: Login screen of Crossroads II Game.

If the user has already registered, he will have to use the login screen in order to manage his game rooms. Once logged in, access the application menú (Figure 143).

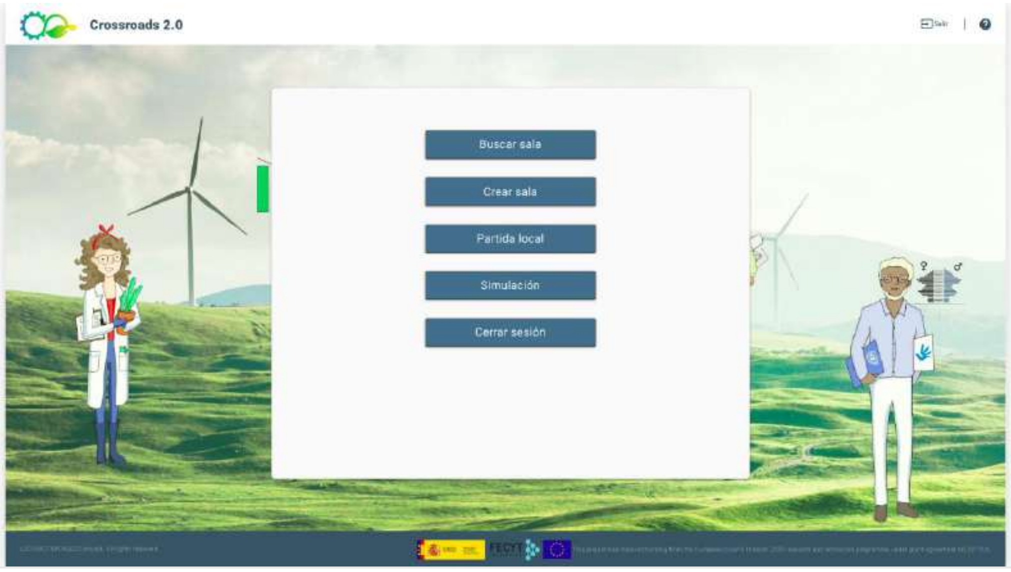


Figure 143: Application menu of Crossroads II Game.

The menu view contains all the actions that a user can perform outside the game. When playing a game, the main option is found in the ``Create room'' button, because when clicking on this button, the room creation process starts, by means of the page shown in the following figure. Other important options are ``Search room'', which shows a history of the rooms created by the user, and the option ``Logout''.

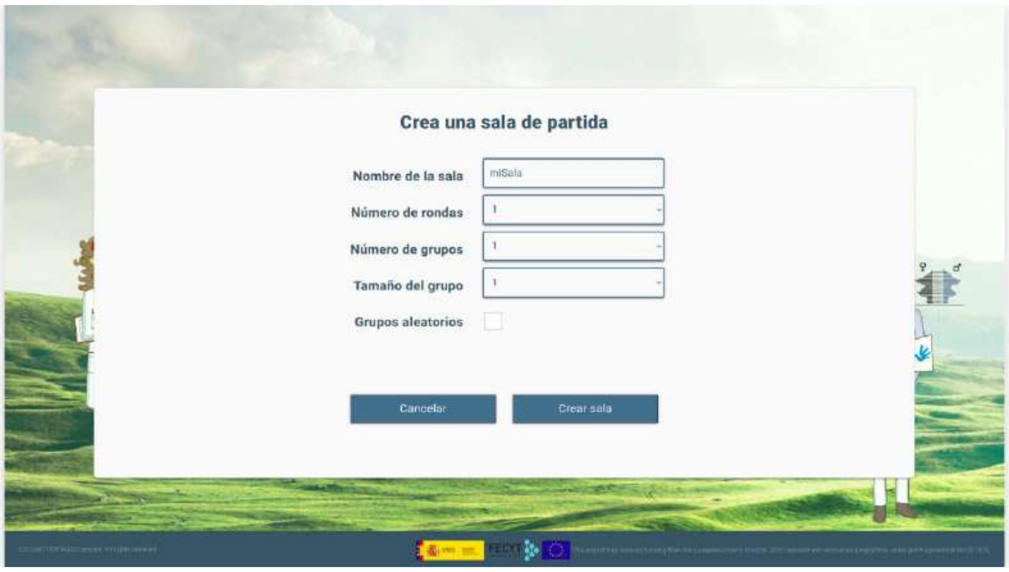
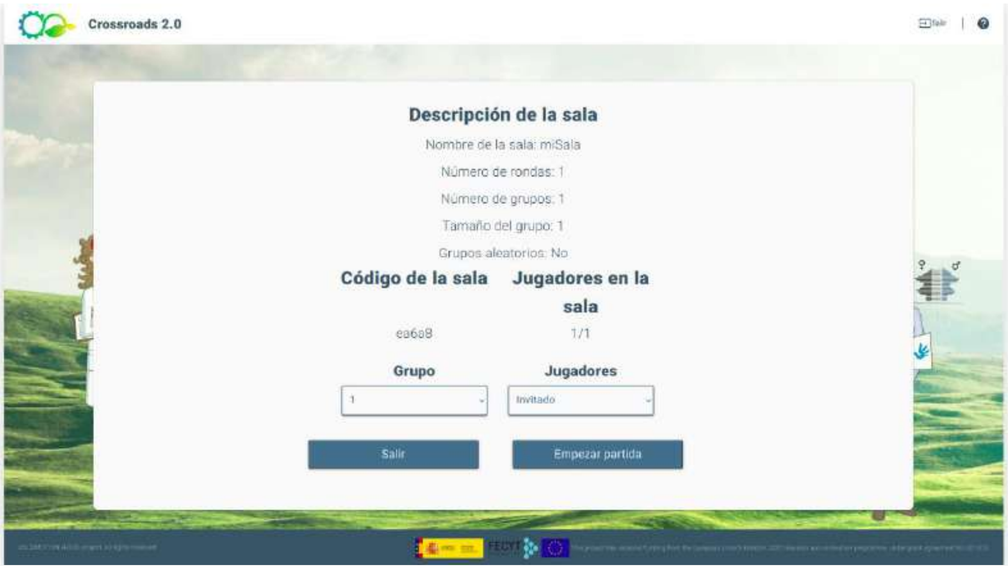


Figure 144: Room creation menu of Crossroads II Game.

On the room creation page, the moderator has to select the room name, the number of rounds, the number of groups and the group size (Figure 144). By clicking on the ``Create room'' button, the data will be validated and the waiting room will be displayed (figure below). If, on the other hand, the moderator selects the ``Cancel'' button, he/she will return to the menu screen.


Crossroads 2.0

Descripción de la sala

Nombre de la sala: miSala

Número de rondas: 1

Número de grupos: 1

Tamaño del grupo: 1

Grupos aleatorios: No

Código de la sala	Jugadores en la sala
ea6a8	1/1

Figure 145: Second step of room creation menu of Crossroads II Game.

Through this page, the moderator can view the data of the game room he has created. In addition, using the two drop-down menus at the bottom, he can monitor the players entering the room and the group they are joining. By clicking on the button at the bottom (``Start game''), the application checks that all groups have at least one player, thus starting the first round of the game and showing the moderator the game control room (Figure 146).

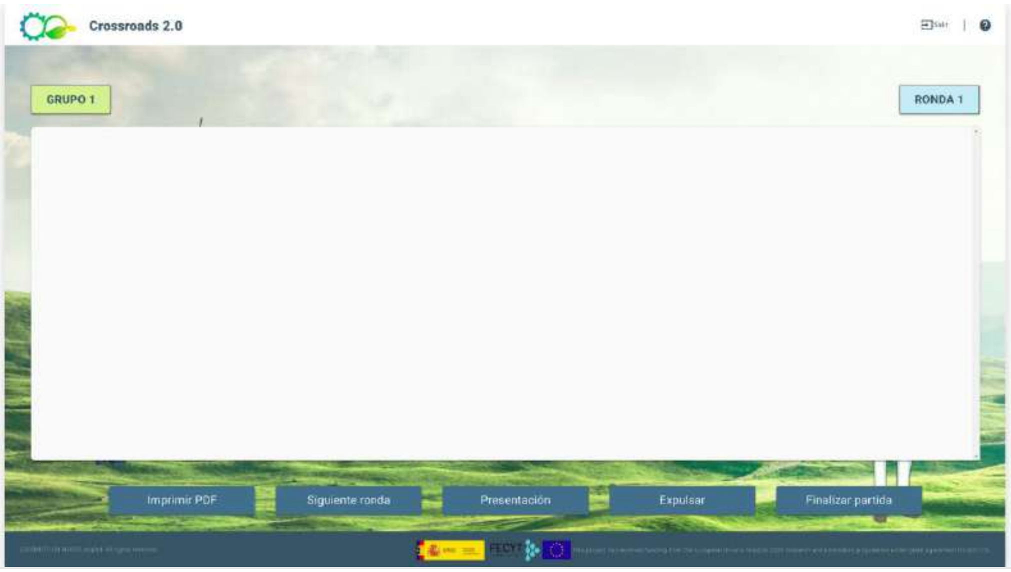
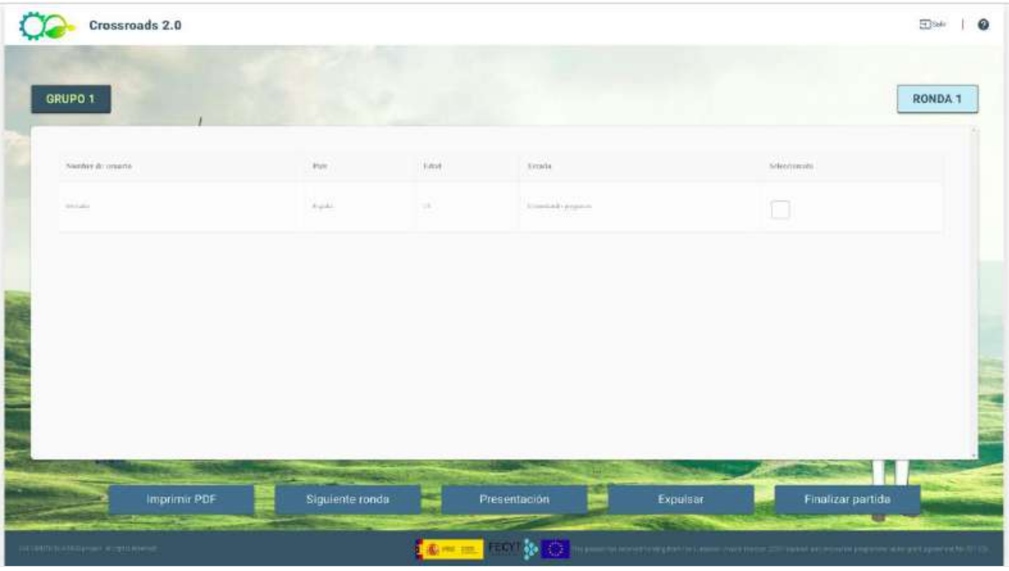


Figure 146: Game monitoring screen.

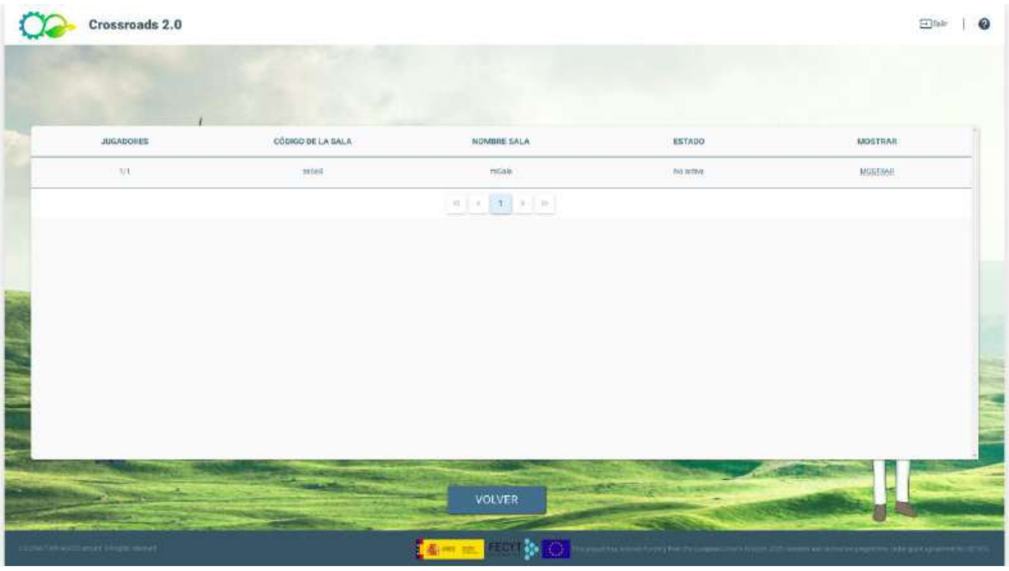
Through the monitoring screen, the moderator can control all aspects of the game. All actions that can be performed by the moderator can be found in the buttons on this page, although the ``EXPEL'' and `` PRESENT'' options are not implemented. The ``NEXT ROUND'' button allows the completion of the current round and, if there are more rounds to play, starts a new round, while the ``END'' button ends the game (Figure 147).



Nombre de usuario	País	Edad	Estado	Seleccionado
[illegible]	España	[illegible]	Esperando jugadores	☐

Figure 147: Game monitoring screen when the moderator selects group 1.

In the moderator's control room, we can monitor the work of the group (Figure 147). If in the control room we select any of the buttons corresponding to the groups, the system will show in the central box a table with the details of the data of each and every one of the players belonging to the group that has just been selected. The checkbox that appears at the end of each row is to select a player and perform some action on him, although at the moment there is none implemented.



JUGADORES	CÓDIGO DE LA SALA	NOMBRE SALA	ESTADO	MOSTRAR
1/1	[illegible]	miSala	No activa	MOSTRAR

Figure 148: Game monitoring screen when the moderator selects round 1.

In the moderator's control room (Figure 148), rounds can also be monitored. By selecting one of the buttons corresponding to the rounds, the system will display in the central table the data related to the status of the round, as well as a table showing the status of each group during that round. The application only allows to display the data of the rounds that have been started, displaying an error message in case you want to see the data of a round.

- PLAYER MANUAL

To join a room and start a game, the player must enter the room code in the form and click on the enter button. If the room is on hold, the player will be able to enter and will be shown a form to enter his profile data (Figure 149).

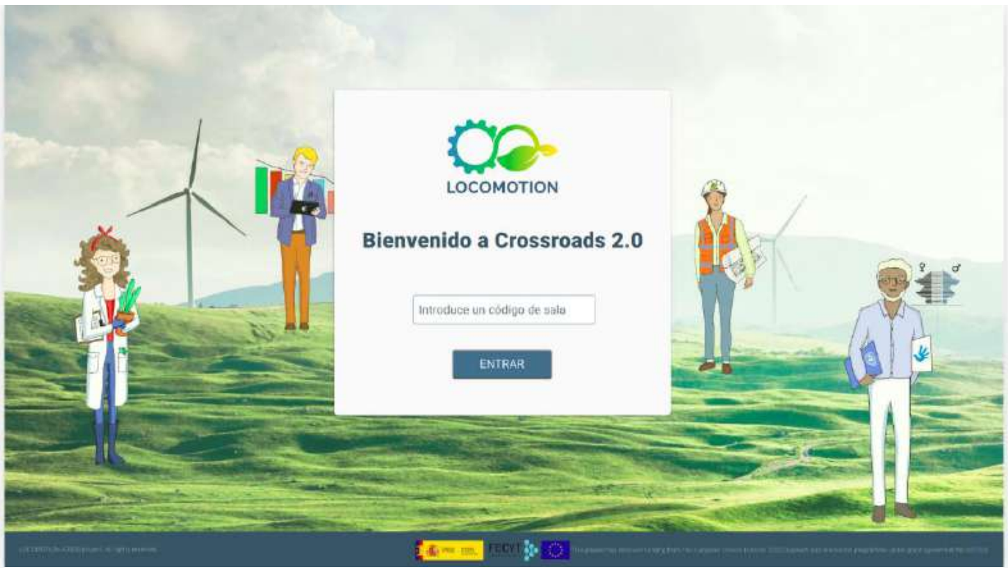


Figure 149: Room connection screen for players.

Figure 150, present the form in which the player has to enter his profile data for the game. As in the rest of the forms of the application, a message with the format of the entry of each field is displayed when hovering the cursor over the field in question. Once the profile data has been entered, a screen with information about the game will be displayed.

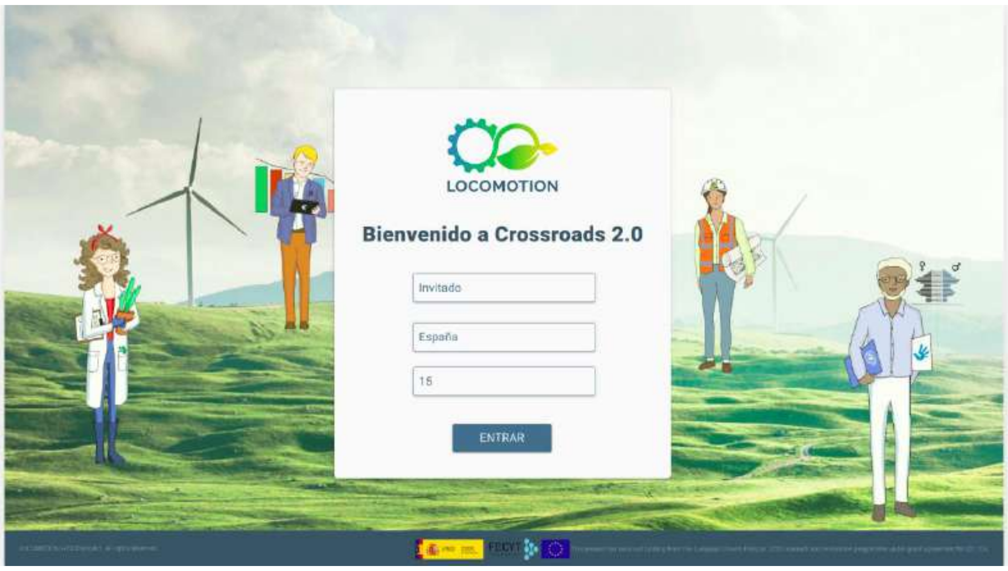


Figure 150: Screen where the player enters his data.

Game information screen displays a video with information about the game (Figure 151). Selecting the option ``I already know how to play'' will load the players' waiting room.

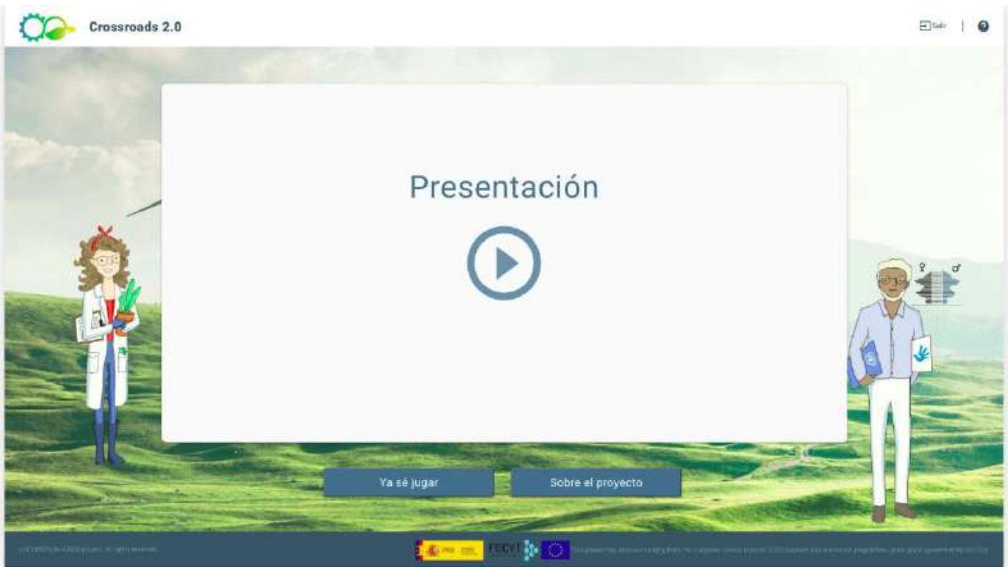
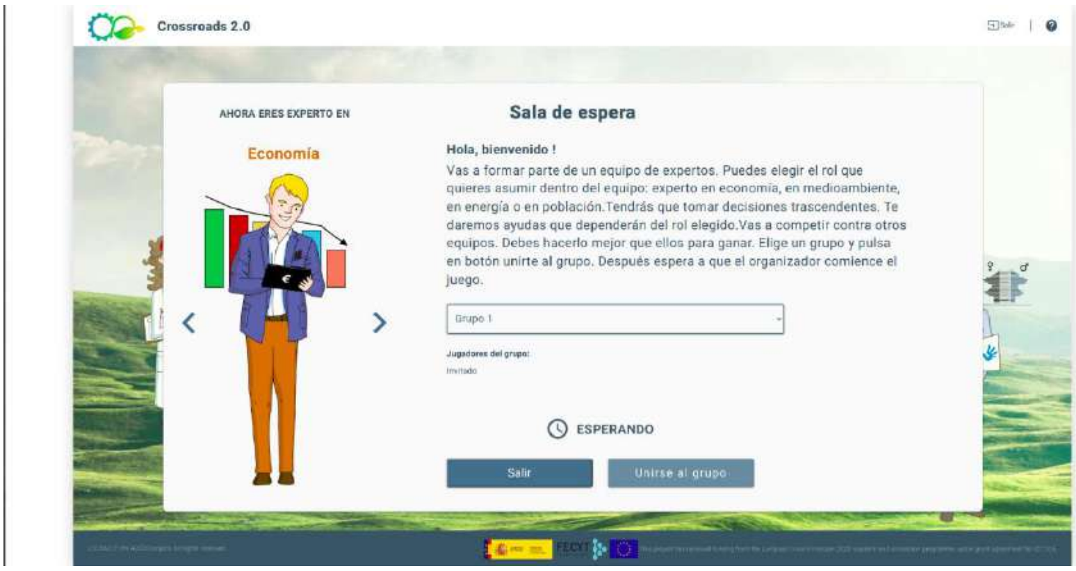


Figure 151: Game information screen for players.

In the waiting room for the players (Figure 152), each player can look at the groups that are there and the players that are in each group by means of the drop-downs. As for the character selector on the left side, it is still unimplemented, so the player does not have to perform any action on it. Once the player has selected the group he wants to join through the first drop-down, he has to click on the ``Join group'' button. If the group has space to spare, the player will have joined properly, so he will have to wait for the moderator to start the game, at which time the questions he will have to answer will be loaded.



AHORA ERES EXPERTO EN

Economía

Sala de espera

Hola, bienvenido !

Vas a formar parte de un equipo de expertos. Puedes elegir el rol que quieres asumir dentro del equipo: experto en economía, en medioambiente, en energía o en población. Tendrás que tomar decisiones trascendentes. Te daremos ayudas que dependerán del rol elegido. Vas a competir contra otros equipos. Debes hacerlo mejor que ellos para ganar. Elige un grupo y pulsa en botón unirte al grupo. Después espera a que el organizador comience el juego.

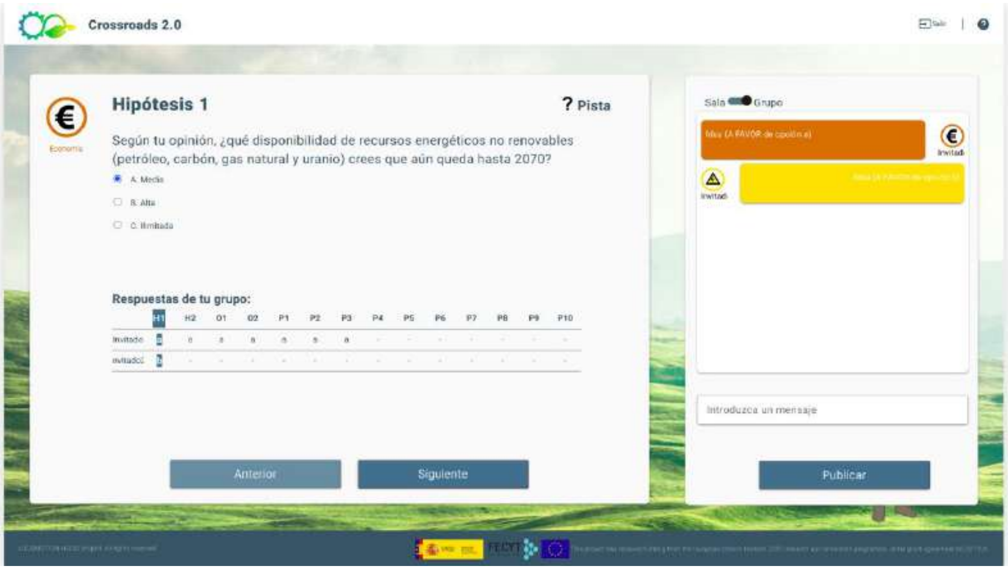
Grupo 1

Jugadores del grupo:

Invitado

Figure 152: Image of the players' waiting room.

The questions page (Figure 153) includes the form that the player has to answer together with his group to elaborate a proposal. When clicking on the ``Next'' button to submit the answer, the player will be asked for an argument to support his choice. The page also allows you to navigate backwards through the form using the ``Back'' button. Once all the questions have been answered, the player will be taken to a page where he/she can see the answers he/she has selected and those selected by the other members of the group.



Hipótesis 1

Según tu opinión, ¿qué disponibilidad de recursos energéticos no renovables (petróleo, carbón, gas natural y uranio) crees que aún queda hasta 2070?

[✓ A. Media]
B. Alta
C. Ilimitada

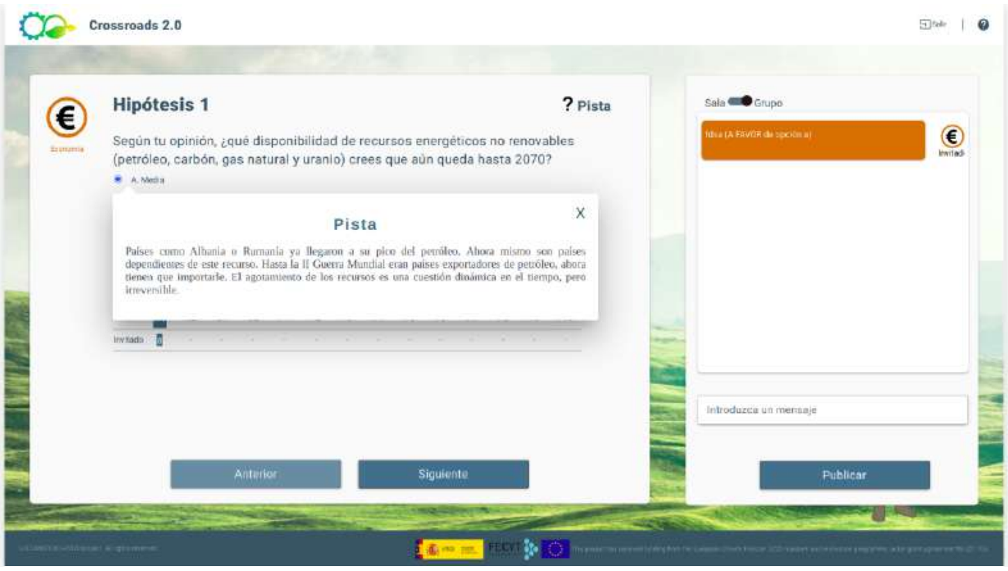
Respuestas de tu grupo:

	H1	H2	O1	O2	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Invitado	[illegible]	8	8	8	8	8	8	-	-	-	-	-	-	-
Invitado	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-

Mas (A FAVOR de opción a)

Figure 153: Screenshot of the questions to be answered by the player.

When submitting the answer to a form question, a pop-up appears asking for an argument to be entered (Figure 154). If no argument is entered, the answer will not be saved.



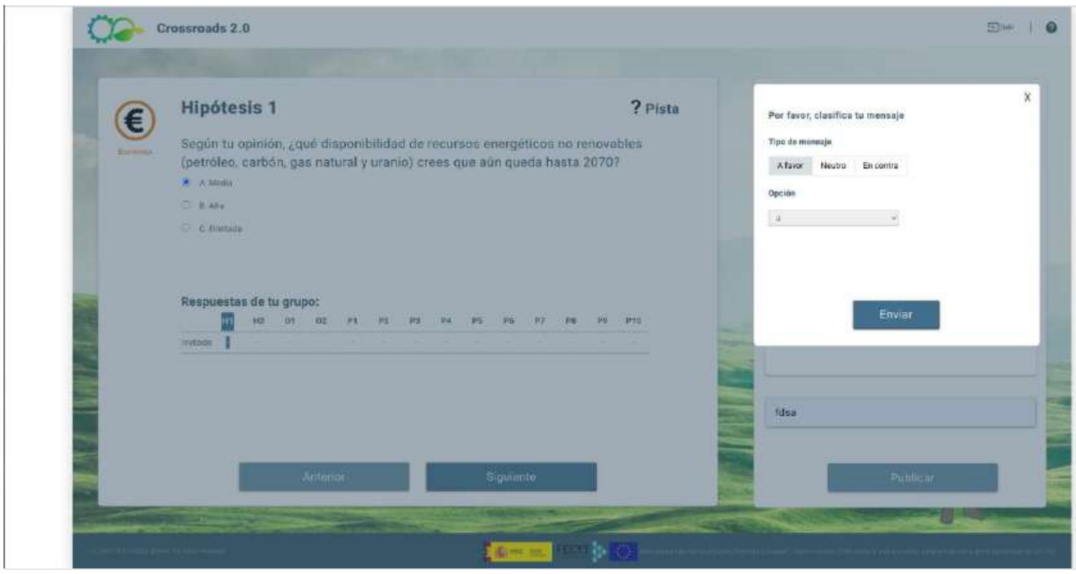
Según tu opinión, ¿qué disponibilidad de recursos energéticos no renovables (petróleo, carbón, gas natural y uranio) crees que aún queda hasta 2070?

Pista

Países como Albania o Rumania ya llegaron a su pico del petróleo. Ahora mismo son países dependientes de este recurso. Hasta la II Guerra Mundial eran países exportadores de petróleo, ahora tienen que importarlo. El agotamiento de los recursos es una cuestión dinámica en el tiempo, pero irreversible.

Figure 154: Image of the question screen with the pop-up requesting the argument.

When you send an argument for an answer, the argument is added to the group chat message list so that other group members can see it to facilitate discussion (Figure 155). The chat also allows you to send messages without having to answer a question. To do this, enter the body of the message and click the ``Post'' button. In response to this action, the system will display a pop-up screen in which the user will have to select the type of message and the question and answer to which the message he/she has just entered refers.

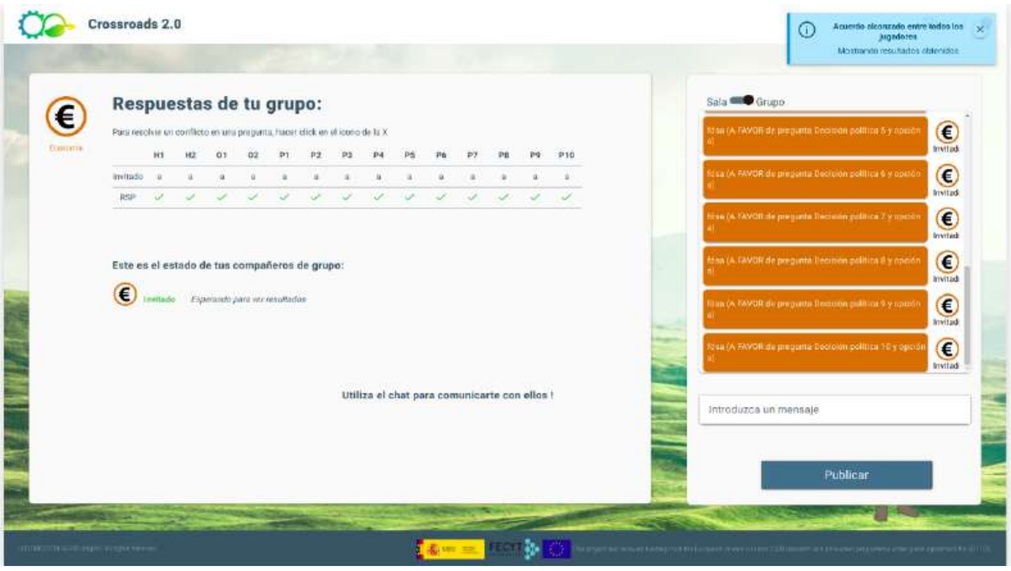


Según tu opinión, ¿qué disponibilidad de recursos energéticos no renovables (petróleo, carbón, gas natural y uranio) crees que aún queda hasta 2070?

Por favor, clasifica tu mensaje

Figure 155: Image of the question screen with the argument of the first question dumped into the chat.

When all the game options have been completed, the possible conflicts are checked (Figure 156). This screen shows the status of each player's form, with the answers that each player has selected and the existence or not of a conflict in each question. It also shows the status of each player at a given time. The player will have to wait to see the round results either until there are no conflicts or until the moderator finishes the round and the conflicts are automatically resolved.


Crossroads 2.0

Acuerdo alcanzado entre todos los jugadores
Mostrando resultados obtenidos

Respuestas de tu grupo:

Para resolver un conflicto en una pregunta, hacer click en el icono de la X

	H1	H2	O1	O2	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
invitado	a	a	a	a	a	a	a	a	a	a	a	a	a	a
RSP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Este es el estado de tus compañeros de grupo:

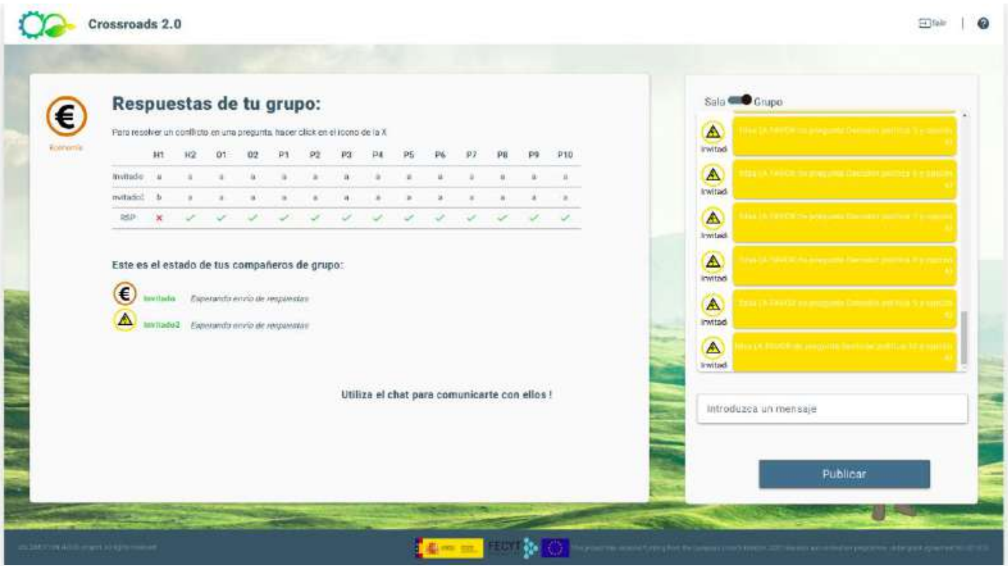
invitado — Esperando para ver resultados

Utiliza el chat para comunicarte con ellos !

fijsa (A FAVOR de pregunta Decisión política 5 y opción a) — invitad
fijsa (A FAVOR de pregunta Decisión política 6 y opción a) — invitad
fijsa (A FAVOR de pregunta Decisión política 7 y opción a) — invitad
fijsa (A FAVOR de pregunta Decisión política 8 y opción a) — invitad
fijsa (A FAVOR de pregunta Decisión política 9 y opción a) — invitad
fijsa (A FAVOR de pregunta Decisión política 10 y opción a) — invitad

Figure 156: Conflict check screen image (no conflicts).

If a conflict arises after the players enter answers to the questions, the conflict will be displayed on this screen with the corresponding icon (Figure 157). In order to resolve the conflict, the player can click on the icon of the question with the conflict, which will open the conflict resolution page for that question.


Crossroads 2.0

Respuestas de tu grupo:

Para resolver un conflicto en una pregunta, hacer click en el icono de la X

	H1	H2	O1	O2	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
invitado	a	a	a	a	a	a	a	a	a	a	a	a	a	a
invitado2	b	a	a	a	a	a	a	a	a	a	a	a	a	a
RSP	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

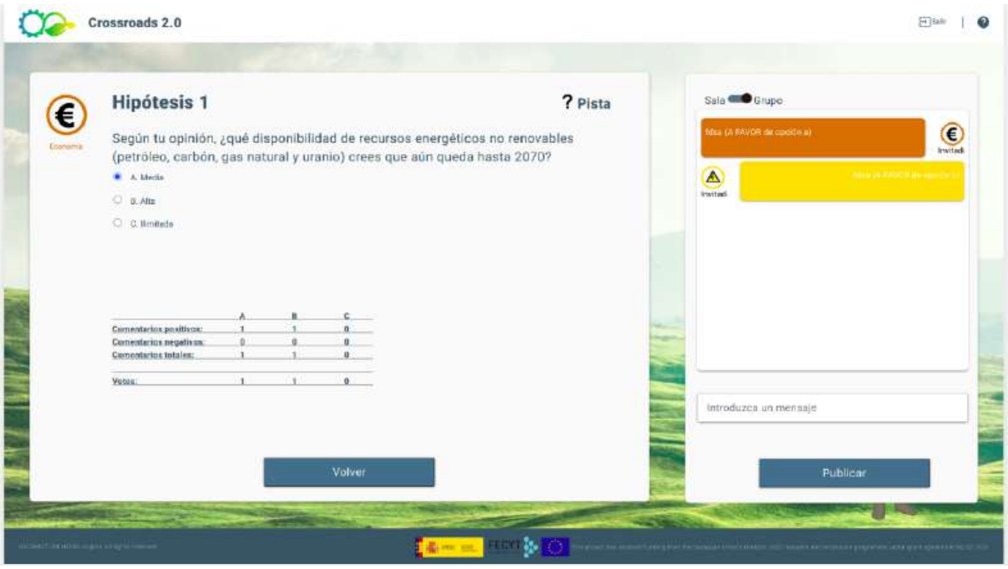
Este es el estado de tus compañeros de grupo:

invitado — Esperando envío de respuestas
invitado2 — Esperando envío de respuestas

Utiliza el chat para comunicarte con ellos !

Figure 157: Conflict check screen image (with conflict in question 1).

In the conflict resolution screen, the form for the question on which you want to resolve the conflict is displayed (Figure 158). In addition, information about the arguments and votes received by each option of the question is provided. Once the answer has been modified (or the action cancelled), navigation to the previous screen takes place.



Hipótesis 1

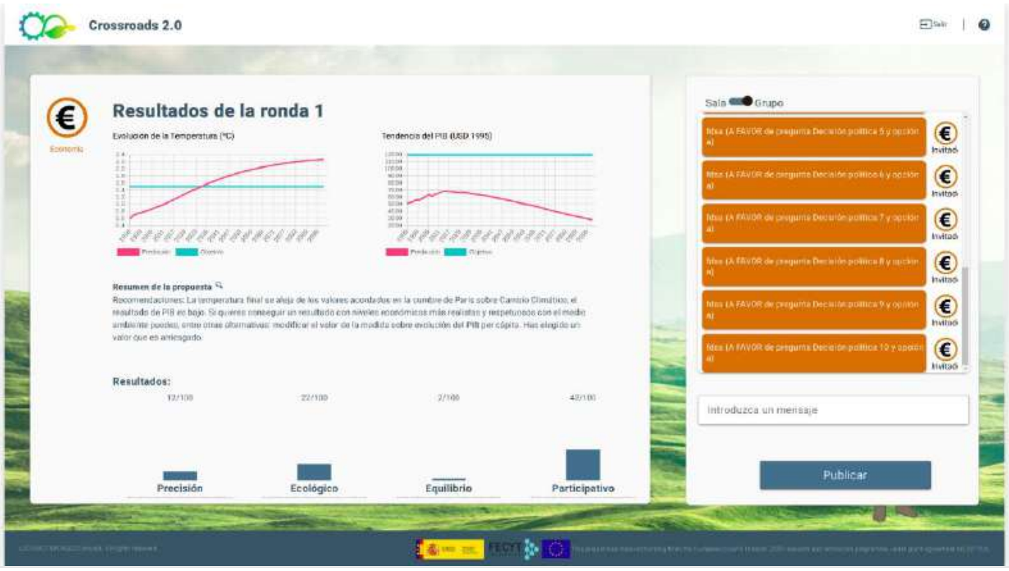
Según tu opinión, ¿qué disponibilidad de recursos energéticos no renovables (petróleo, carbón, gas natural y uranio) crees que aún queda hasta 2070?

[✓ A. Media]
B. Alta
C. Ilimitada

	A	B	C
Comentarios positivos:	1	1	0
Comentarios negativos:	0	0	0
Comentarios totales:	1	1	0
Votos:	1	1	0

Figure 158: Conflict Screen shot of the conflict resolution screen for question 1.

The results screen shows the results of the round for the group (Figure 159). It includes the graphs with the evolution of GDP and temperature as a consequence of the selected measures. It also includes a recommendation message to improve the results obtained and the scores achieved by the group in that round. The player will remain on this screen until the moderator starts a new round or until the moderator ends the game, at which time the final results of the game will be displayed.


Crossroads 2.0
Economía
Evolución de la Temperatura (°C)
Tendencia del PIB (USD 1995)
Predicción
Objetivo
Sala Grupo
Idea (A FAVOR de pregunta Decisión política 5 y opción a)
Idea (A FAVOR de pregunta Decisión política 6 y opción a)
Idea (A FAVOR de pregunta Decisión política 7 y opción a)
Idea (A FAVOR de pregunta Decisión política 8 y opción a)
Idea (A FAVOR de pregunta Decisión política 9 y opción a)
Idea (A FAVOR de pregunta Decisión política 10 y opción a)
Invitado
Introduzca un mensaje
Publicar

Resultados de la ronda 1

Resumen de la propuesta

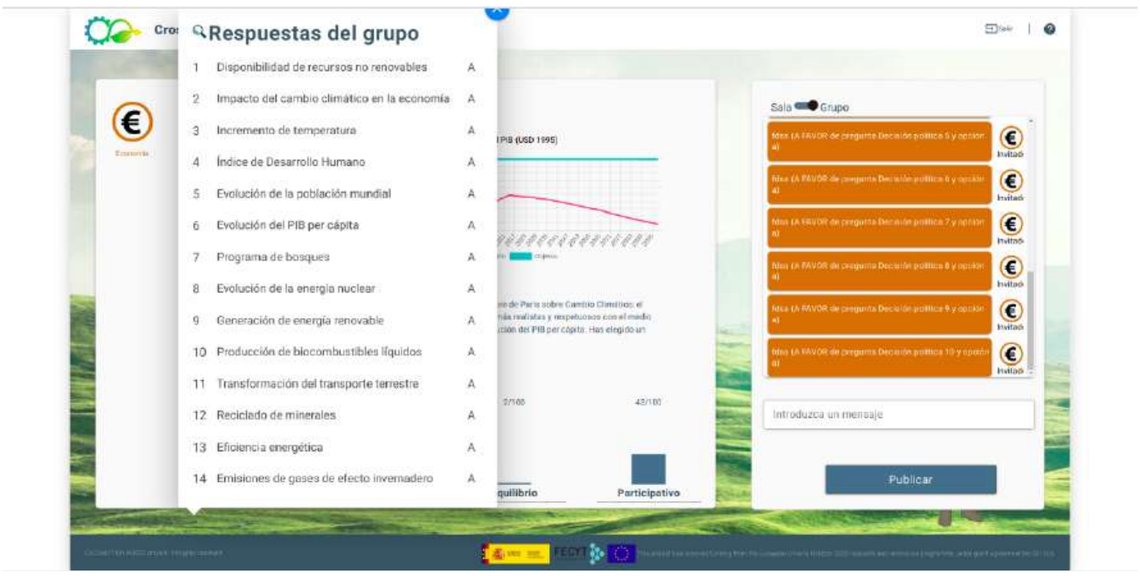
Recomendaciones: La temperatura final se aleja de los valores acordados en la cumbre de Paris sobre Cambio Climático, el resultado de PIB es bajo. Si quieres conseguir un resultado con niveles económicos más realistas y respetuosos con el medio ambiente puedes, entre otras alternativas: modificar el valor de la medida sobre evolución del PIB per cápita. Has elegido un valor que es arriesgado.

Resultados:

Precisión	Ecológico	Equilibrio	Participativo
12/100	22/100	2/100	42/100

Figure 159: Conflict Results screen image for round 1.

The group response screen under Round Results shows a final report of the game performance. If we click on the magnifying glass icon in the round results screen, a pop-up containing the proposal responses submitted by the group will be displayed (Figure 160).



Respuestas del grupo

1	Disponibilidad de recursos no renovables	A
2	Impacto del cambio climático en la economía	A
3	Incremento de temperatura	A
4	Índice de Desarrollo Humano	A
5	Evolución de la población mundial	A
6	Evolución del PIB per cápita	A
7	Programa de bosques	A
8	Evolución de la energía nuclear	A
9	Generación de energía renovable	A
10	Producción de biocombustibles líquidos	A
11	Transformación del transporte terrestre	A
12	Reciclado de minerales	A
13	Eficiencia energética	A
14	Emisiones de gases de efecto invernadero	A

Sala Grupo

Idea (A FAVOR de pregunta Decisión política 5 y opción a) Invitado

Idea (A FAVOR de pregunta Decisión política 6 y opción a) Invitado

Idea (A FAVOR de pregunta Decisión política 7 y opción a) Invitado

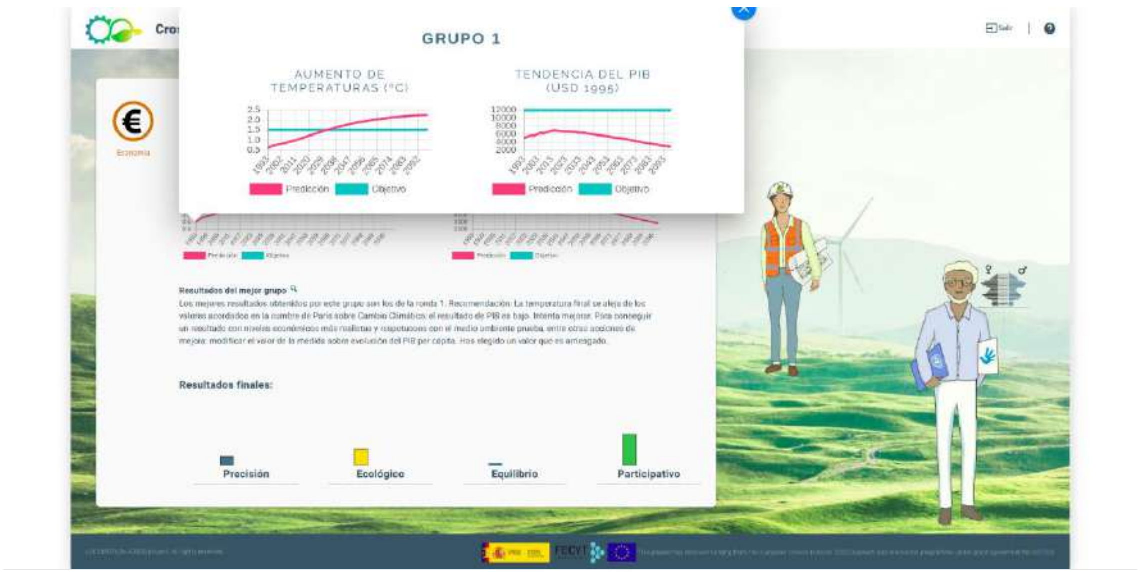
Idea (A FAVOR de pregunta Decisión política 8 y opción a) Invitado

Idea (A FAVOR de pregunta Decisión política 9 y opción a) Invitado

Idea (A FAVOR de pregunta Decisión política 10 y opción a) Invitado

Figure 160: Image of the final results screen.

In the final results screen, the final results of the game for the group are displayed. It includes the graphs of the round in which the group scored best on the ``Balance'' scale. It also includes a podium with the three best groups in each of the scores (Figure 161).



Resultados del mejor grupo

Los mejores resultados obtenidos por este grupo son los de la ronda 1. Recomendación: La temperatura final se aleja de los valores acordados en la cumbre de París sobre Cambio Climático; el resultado de PIB es bajo. Intenta mejorar. Para conseguir un resultado con niveles económicos más realistas y respetuosos con el medio ambiente prueba, entre otras acciones de mejora: modificar el valor de la medida sobre evolución del PIB per cápita. Has elegido un valor que es arriesgado.

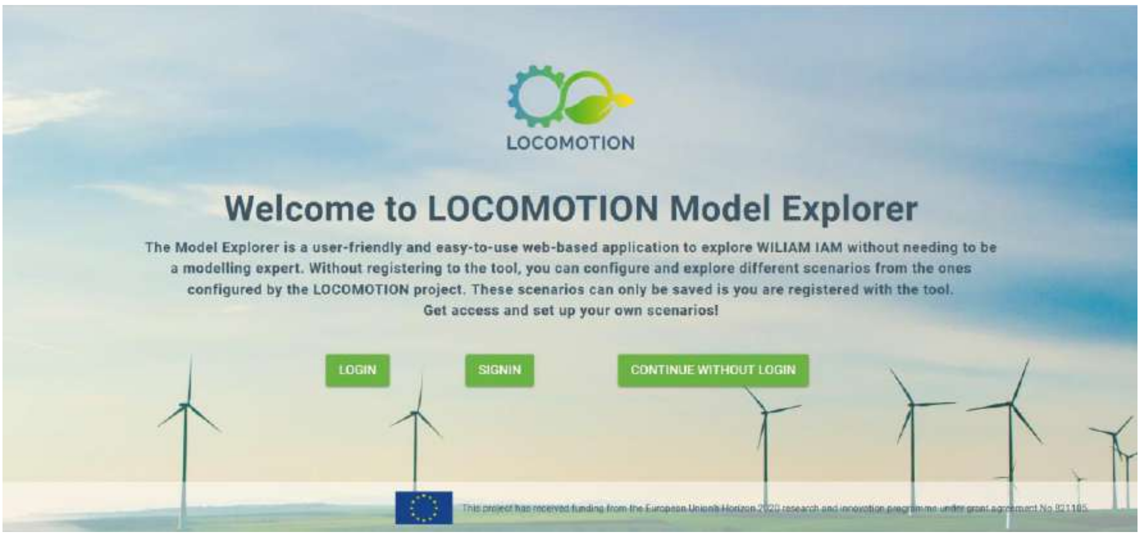
Resultados finales:

Figure 161: Image of the final results screen with the pop-up of the graphs of the best group.

In the graphs of the best group, if the player wants to see the graphs corresponding to the final results of the best group, just click on the magnifying glass button on the final results screen. This action will display a drop-down with the two graphs of that group.

MODEL EXPLORER

Once you click in the tool link provided in the website, the user is re-directed to the homepage of the Model Explorer (Figure 162) where three different alternatives are provided: *login* with your credentials (only for registered users), *sign in* as a new tool user or *continue without login*. These alternatives are related with the functionalities that the tool will provide users. Only registered users are able to save scenario after the simulation in the tool. If you are not registered, you can visualize results losing them when you change to another view or re-load the page.



Welcome to LOCOMOTION Model Explorer

The Model Explorer is a user-friendly and easy-to-use web-based application to explore WILIAM IAM without needing to be a modelling expert. Without registering to the tool, you can configure and explore different scenarios from the ones configured by the LOCOMOTION project. These scenarios can only be saved is you are registered with the tool.
Get access and set up your own scenarios!

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821105.

Figure 162: Main page of Model Explorer tool.

If you click on *signin*, a new view is opened where the user is able to create a new user account by means of the email, user name and password (Figure 163). On the other hand, if you are already registered, you can go directly to login and get access to you user account by means of the user name and password (Figure 164).

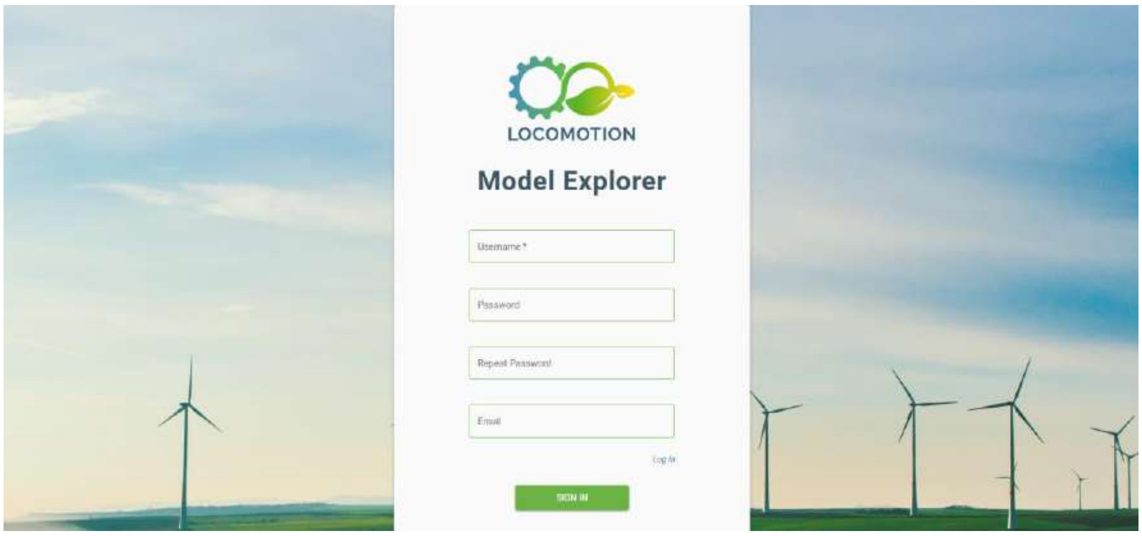


Figure 163: Screenshot of Model Explorer signin.

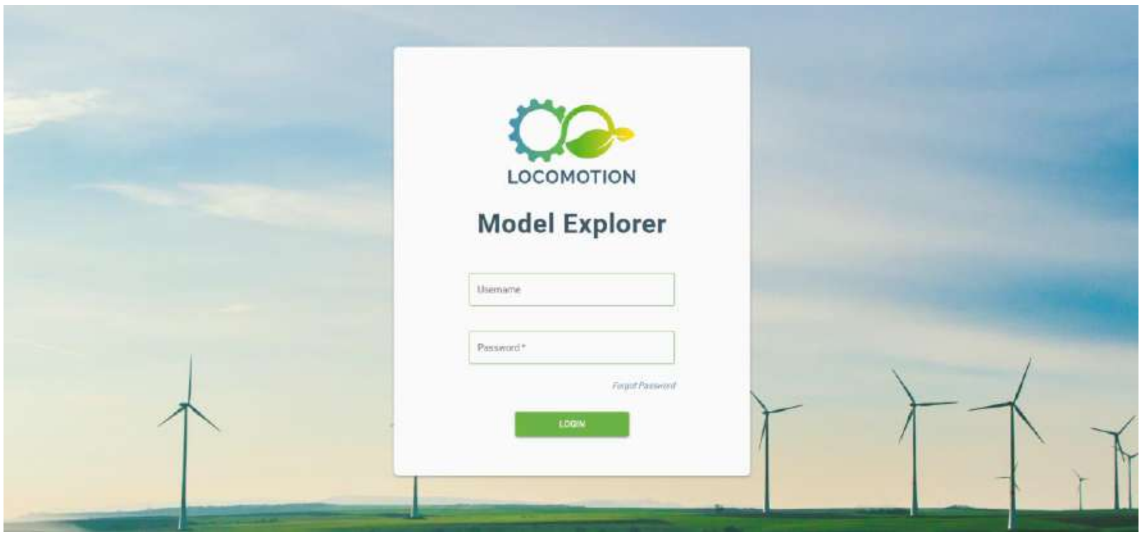


Figure 164: View of Model Explorer login.

In addition to the interfaces for access as a registered user, functionality has also been provided to recover the password account in case the user has forgotten it (Figure 165). This password recovery functionality is very useful to prevent a user from creating a new account every time he forgets his password.

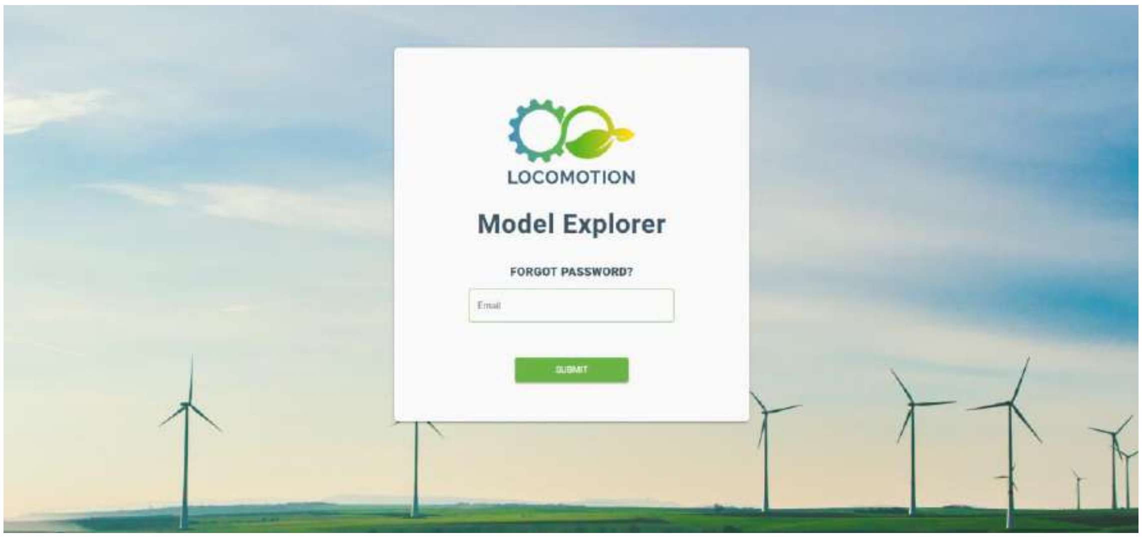
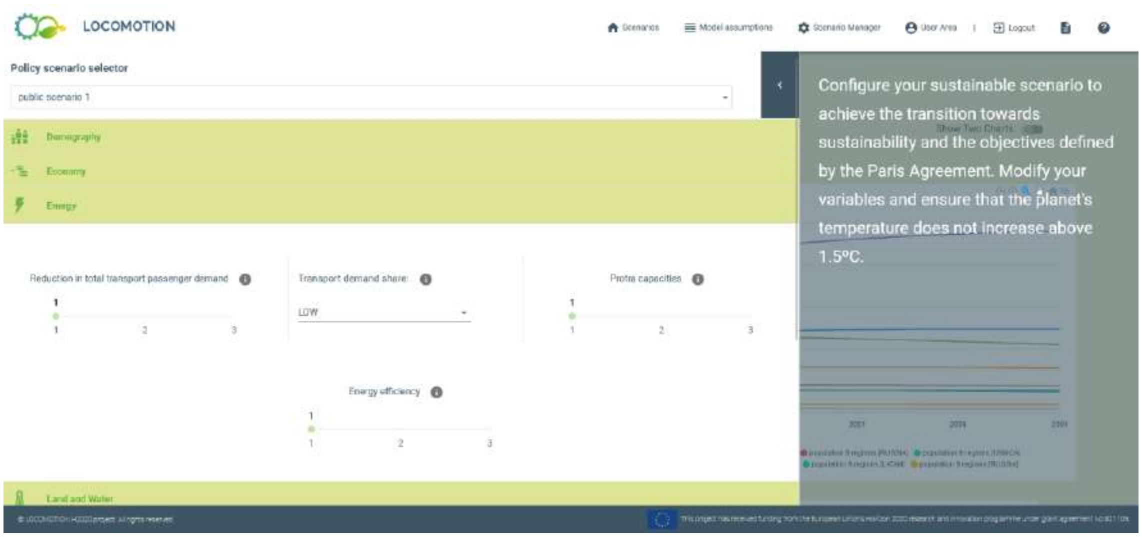


Figure 165: Screenshot of Model Explorer to recover password.

If you access as a non-registered user, the application will redirect you to the main page where you can configure the policies and evaluate the results through 30 indicators, which graphically represent the evolution of the variables along the 2016-2060-time scale. The configuration of the policies is done through the drop-down located on the left of the screen, which opens the policies and the alternatives for their parameterization (Figure 166). After parameterizing the policies, you must validate their application through the apply button which is at the bottom of the screen.



Configure your sustainable scenario to achieve the transition towards sustainability and the objectives defined by the Paris Agreement. Modify your variables and ensure that the planet's temperature does not increase above 1.5ºC.

Figure 166: Screenshot of Model Explorer for scenario parametrization.

When a user is registered in the application, he has a scenario manager where he has the possibility of creating new scenarios. To do this, you need to provide a scenario name and a description (Figure 167). Once the scenario is created, you will be able to access to the main page of the application, parameterize the policies and apply to calculate the indicators. After obtaining the results, at the bottom of the window, the user will have the button to save the scenario. This scenario manager will allow you to view the scenarios already created and even access them again to modify the implemented policies.

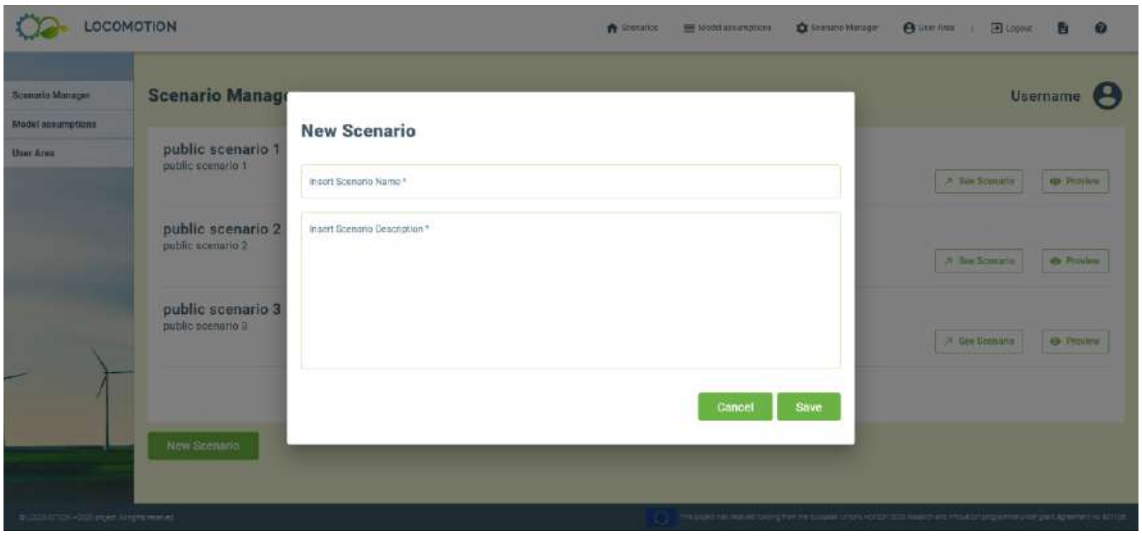


Figure 167: Screenshot of Model Explorer scenario manager a new scenario creation.

The visualization of results will allow you to analyze the evolution of an indicator over time (Figure 168) and comparing indicators through parallel views (Figure 169). The tool also shows the variations in the implemented policies and a global temperature indicator that allows the user to quickly and visually evaluate the results of the selection and application of policies (Figure 170).

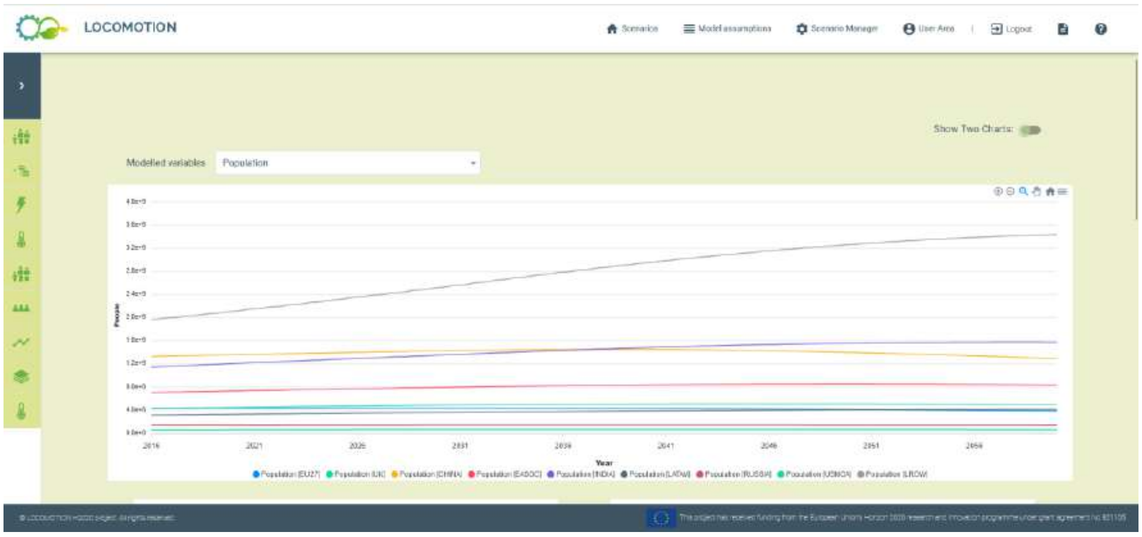


Figure 168: Screenshot of Model Explorer results visualization, one indicator.

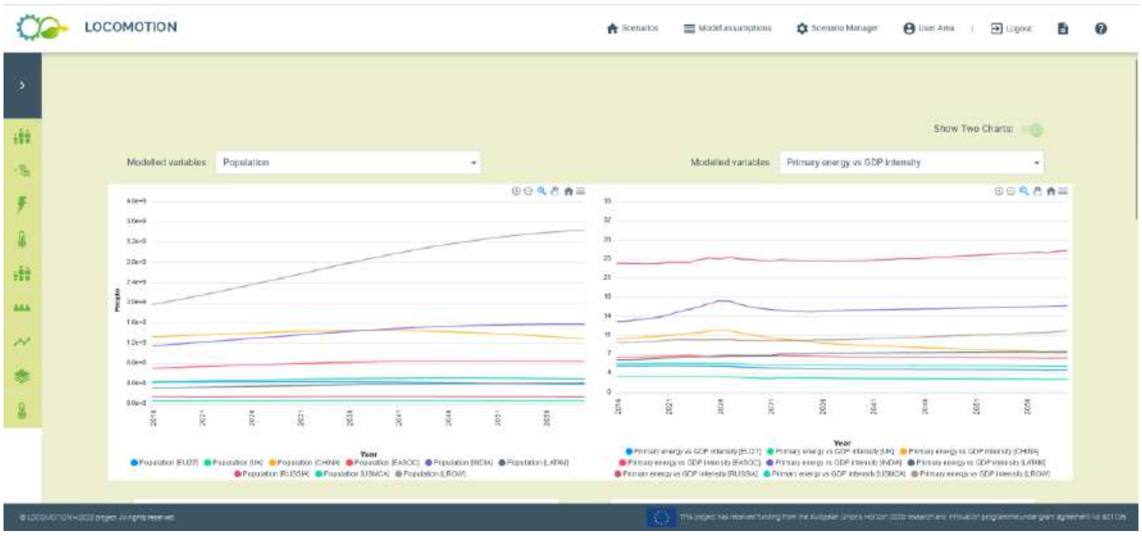
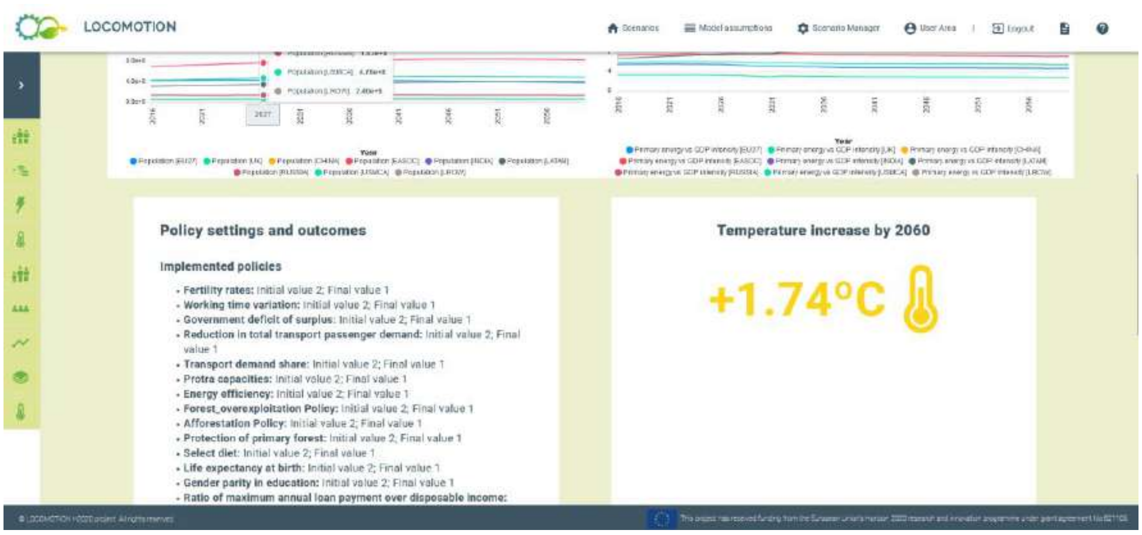

LOCOMOTION
Scenarios
Model assumptions
Scenario Manager
User Area
Logout
Show Two Charts:
Modelled variables
Population
Modelled variables
Primary energy vs GDP intensity
People
Year
Population [EU27]
Population [UK]
Population [CHINA]
Population [EASOC]
Population [INDIA]
Population [LATAM]
Population [RUSSIA]
Population [USMCA]
Population [LROW]
Primary energy vs GDP intensity [EU27]
Primary energy vs GDP intensity [UK]
Primary energy vs GDP intensity [CHINA]
Primary energy vs GDP intensity [EASOC]
Primary energy vs GDP intensity [INDIA]
Primary energy vs GDP intensity [LATAM]
Primary energy vs GDP intensity [RUSSIA]
Primary energy vs GDP intensity [USMCA]
Primary energy vs GDP intensity [LROW]

Figure 169: Screenshot of Model Explorer results visualization, comparative o two indicators.



Policy settings and outcomes

Implemented policies

- **Fertility rates:** Initial value 2; Final value 1
- **Working time variation:** Initial value 2; Final value 1
- **Government deficit of surplus:** Initial value 2; Final value 1
- **Reduction in total transport passenger demand:** Initial value 2; Final value 1
- **Transport demand share:** Initial value 2; Final value 1
- **Protra capacities:** Initial value 2; Final value 1
- **Energy efficiency:** Initial value 2; Final value 1
- **Forest_overexploitation Policy:** Initial value 2; Final value 1
- **Afforestation Policy:** Initial value 2; Final value 1
- **Protection of primary forest:** Initial value 2; Final value 1
- **Select diet:** Initial value 2; Final value 1
- **Life expectancy at birth:** Initial value 2; Final value 1
- **Gender parity in education:** Initial value 2; Final value 1
- **Ratio of maximum annual loan payment over disposable income:**

Temperature increase by 2060

+1.74°C

Figure 170: Screenshot of Model Explorer results visualization, main outcomes and temperature indicator.

Finally, there is a window where the policies that have not remained unchanged during the simulation are presented. These policies were fixed to obtain pre-simulated results of the model and store then to develop the Model Explorer tool. In addition to what is included in this annex, the tool itself includes a small manual for its use.