ENVISAGE

ENhance **VI**rtual learning **S**paces using **A**pplied **G**aming in **E**ducation

H2020-ICT-24-2016

D4.3 – Technology Assembly, Integration and Validation for the implementation of the authoring tool

Dissemination level:	Public (PU)
Contractual date of delivery:	Month 9, 30th June, 2017
Actual date of delivery:	Month 12, 30th Sept., 2017
Workpackage:	WP4 - Virtual labs authoring tool
Task:	T4.3 – Technology Assembly, Integration and Validation for the implementation of the authoring tool
Type:	Report
Approval Status:	Final
Version:	1
Number of pages:	46
Filename:	D4.3_Envisage_AIV_plan.docx

Abstract: This is a report on the integration protocol established between the individual components (game-visual analytics, Unity3D, Web interface), as well as on the integration tests that have been undertaken to verify the operational capacity of each module. The goal of this task is the technical verification of the products of the project with respect to their expected functionality before reaching the prototype level. This task will define a group of internal beta-testers (teachers from EA partner) for the "Virtual labs authoring tool" ensuring that the enabling technology has reached the necessary quality level for supporting the pilots developed in WP5. In order to ensure that the developed solution meets the expected requirements, validation sessions will be put in place. These validation sessions will rely on an Assembly, Integration and Validation (AIV) plan that will be written in parallel with the development tasks. The AIV plan will specify the integration protocol such as fixing the integration interfaces and practices between the individual technology components in the authoring tool. The goal of this protocol is to allow for individual components to work independently on algorithmic refinements and optimizations, while ensuring their smooth integration with the architecture of the authoring tool. Finally, the validation tests will be put in action by the EA partner with teachers in various sciences. In this way, we will verify

that ENVISAGE technologies function to the expected performance criteria in an environment close to that of the author-user.
The information in this document reflects only the author's views and the European Community is not liable for any use that may be made of the information contained therein. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Co-funded by the European Union

Acknowledgment

This work is part of project ENVISAGE that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731900.

Copyright

History

Version	Date	Reason	Revised by
v0.1	31/8/2017	Initial draft with Table of Contents	Dimitrios Ververidis
v0.2	06/9/2017	Test scenarios of front-end	Giannis Chantas
v0.3	15/9/2017	Add scenarios	Dimitrios Ververidis Giannis Chantas
v0.4	21/9/2017	Add analytics details	Fabian Hadiji
v0.5	22/9/2017	Visualization analytics	Christoffer Holmgard
v0.9	03/10/2017	Review	Fabian Hadiji
v1	04/10/2017	Comments addressed	D. Ververidis

Author list

Organization	Name	Contact Information
CERTH	Dimitrios Ververidis (DV)	ververid@iti.gr
CERTH	Stathis Nikolaidis (SN)	stathis.nikolaidis@iti.gr
CERTH	Anastasios Papazoglou - Chalikias (AP)	tpapazoglou@iti.gr
CERTH	Giannis Chantas (GC)	gchantas@iti.gr
UoM	Christoffer Holmgard (CH)	holmgard@gmail.com
GIO	Marc Mueller (MM)	marc@goedle.io
GIO	Fabian Hadiji (FH)	fabian@goedle.io

Executive Summary

This deliverable describes from a high-level perspective how each component is embodied into a single product, and provides instructions on how to test and evaluate each component and all components together. The “Virtual Labs Authoring tool”, which is the main product of ENVISAGE, is a plugin for WordPress, the most popular web content management system. The plugin is a form of software that augments the functionalities of the already available WordPress site into a Virtual Lab authoring platform. The developed plugin has several components such as the front-end interface, the compiling mechanism, and the analytics component. We provide the integration protocol followed and the testing procedures for the functionalities offered for all the components. The instructions focus on to how test and evaluate the components from the scope of the programmer, the educator, and the learner.

All code repositories are publicly available in GitHub under the organization ENVISAGE-H2020. The main code repository is the ‘Virtual-Labs-Authoring-tool’. The link to these repositories is

https://github.com/Envisage-H2020

Evaluation tests are performed in the deployed version of the authoring tool which can be found at:

http://envisage.mklab.iti.gr

username: author

password: reviewerenvisag

Abbreviations and Acronyms

CMS	Content Management System
API	Application Program Interface
GUI	Graphic User Interface
JSON	JavaScript Object Notation
WebGL	Web Graphics Language
SDK	Software Development Toolkit
MAT	Unity3D Material file format
OBJ	Wavefront Object file format
MTL	Wavefront Material file format
LMB	Left Mouse button
RMB	Right Mouse button
Breadcrumbs	Breadcrumbs or breadcrumbs trail is a graphical control element used as a navigational aid in user interfaces. Example : Home ▸ Project Editor ▸ 3D Scene Editor

Table of Contents

1. Integration activities and testing protocol

In this section, we provide the Assembly, Integration, Validation (AIV) plan for the "Virtual Labs Authoring tool", which we have named as WordpressUnity3DEditor for the time being. Details about the implementation architecture of WordpressUnity3DEditor can be found in D4.2. The AIV plan consists of details about how each software component is integrated in the WordpressUnity3DEditor; and details about how each software component can be validated, i.e. how to check that each component is working properly.

1.1 Integration activities

The latest developments towards an integration are categorized into three parts, namely front-end integration, back-end compiling, and analytics integration. As with regards to the front-end integration, we have developed a user-friendly interface dedicated to the needs of authoring a Virtual Lab. The front-end was developed from scratch taking into consideration the first comments of the consortium in the previous plenary meeting and later from their hand-written evaluations. The effort resulted in a modern look, fast, and clear graphic interface with many asynchronous task capabilities that do not require the re-loading of the whole page.

Back-end compiling was improved with several architectural developments that allow the logical split of a Game Project into entities. That is, Scenes and Assets of a Game Project were redefined in a more professional manner. Additional actions were developed that resulted into a fully operational Virtual Labs authoring tool consisting of the main 3 functionalities, namely create, edit, delete all 3 entities, that is, game projects, scenes, and assets. The Wind Energy simulation game was split into pieces of code that allow the reproduction of several Wind Energy Games with various assets and scenes. The educators can upload their own 3D models, define the properties of the terrain, the energy consumers, the energy producers, and add decoration assets in a scene without any programming. The compiling was developed and tested for both Windows and Linux based servers, two technologies that cover 99% of the market[1]. The compiling is based on Unity3D game engine that provides free and proprietary tools for making professional games. The export game formats are for Web browsers as well as for Windows, Mac, and Linux standalone desktop binary formats covering a great percentage of the market.

With regards to the Analytics integration, the Virtual labs plugin was connected with the analytics server through an iframe technology that allows code to be integrated, visualized, and maintained from the analytics server. The iframe is presented in a tab next to each game scene that allows the educator to see game analytics filtered by country, action, and time intervals.

The results of the integration activities are shown in the testing plan that is presented in the following lines. The testing plan consists of usage scenarios and the expected behavior. Declinations from the expected behavior are issues and bugs that should be reported according to a specified protocol as follows.

1.2 Testing Protocol

The testing protocol distinguishes three categories of users as shown in Table 1, namely a)

[1] https://en.wikipedia.org/wiki/Usage_share_of_operating_systems

the programmer that performs the installation and configuration actions; b) the advanced educator that creates/edits/deletes the games/scenes/assets; b) the simple educator that performs the create/edit of the scenes and c) the learner that plays the games.

Table 1: User categories

	a) **Programmer**: Someone that installs and sets up the system with basic knowledge in operating systems, server technologies, and Web Content Management Systems (CMS).
	b) **Advanced educator** : This type of user can perform all actions in the game, i.e. create/edit/delete a game, a scene or an asset. b) **Educator** : This user can perform only scene creation or editing.
	c) **Learner:** The receiver of the knowledge in the form of a game.

The tests to be done are summarized in Table 1. All these tests are included in scenarios, each one being a bundle of actions.

Table 1: Test plan, summary of tests

#	Action	Scope	Definition	Scenario
1. Test Installation				
1.1	Test installation of Unity in Linux	Programmer	Run Unity in Linux and verify it is working properly.	Scenario 0
1.2	Test installation of Unity in Windows	Programmer	Run Unity in Windows and verify it is working properly.	Scenario 0
1.3	Installation of WordpressUnity3DEditor	Programmer	Install our plugin in WordPress.	Scenario 0
1.4	Test remote compiling in Linux	Programmer	Compile a test game in a Ubuntu based server.	Scenario 0
1.5	Test remote compiling in Windows	Programmer	Compile a test game in a Windows based server.	Scenario 0
1.6	Test data integration	Programmer	Test integration for sending data to the tracking infrastructure.	Scenario 0

1.7	Test Analytics	Programmer	Is the plugin connected to analytics server correctly.	Scenario 0
2. Test content creation				
2.1	Create a Game Project	Educator	A Game project should be created in order to make a Game.	Scenario 1
2.2	Create a Scene	Educator	Each Game Project can have multiple scenes.	Scenario 1
2.3	Create an Asset	Educator	Each Game Project should have many Assets available for inserting into the Game's Scenes.	Scenario 1
2.4	Edit a 2D Scene	Educator	A 2D scene is a GUI that should be editable. Such scenes are the Main Menu, Credits, Options, Login etc.	Scenario 1
2.5	Edit a 3D Scene	Educator	A 3D scene should be editable with a 3D editor.	Scenario 1
2.6	Edit an Asset	Educator	Assets should be editable after their creation.	Scenario 3
2.7	Delete a Game Project	Educator	The entire game project should be deleted. Its assets and scenes should be also deleted.	Scenario 3
2.8	Delete a Scene	Educator	A scene of the Game Project should be deleted.	Scenario 3
2.9	Delete an Asset	Educator	An Asset of the Game Project should be deleted.	Scenario 3
2.3.1	Uploading 3D content	Educator	Upload objects, textures, and materials to asset3D.	Scenario 2
2.3.2	Uploading 2D content	Educator	Upload images as jpgs to asset3D.	Scenario 2
2.4	Using Analytics	Educator	How to interpret analytics.	Scenario 3
3. Test game compiling				
3.1	Generate Games at Windows server	Educator	Send preferences to server to compile the game.	Scenario 4
3.2	Generate Games at Linux server	Educator	Send preferences to server to compile the game.	Scenario 4
3.3	Test WebGL generated games	Educator	Is the link to play functional? How to download and install the game to another server?	Scenario 4

4. Test the game				
4.1	Playing the VR Lab Games	Learner	How to ensure that the game behavior is the expected one.	Scenario 5

The aforementioned tests should be evaluated according to the template shown in Table 1. In "Software identification", some details about which component is tested should be provided. In "Test Period", details about the time of the test and the person who is testing it should be provided. In "Test Environment" the systems used for testing should be described. In references, any existing web link for the problem should be given. In "Scenarios", the action to be made and the expected behavior should be described. In "Issues Raised", the bugs or required changes should be provided.

TEMPLATE FOR THE TEST EVALUATIONS

Software identification	
Name	Name of the component to test e.g. 2.3
Versions	[Version is stated in Wordpress -> Plugins -> WordpressUnity3DEditor]

Test period	
Test phase	1st or 2nd
Test Types	[Functional / Interoperability / Compliance / User Acceptance]
Test Status	Test Completed
Planned test start date	YY/XX/2017
Actual test start date	YY/XX/2017
Test completion date	YY/XX/2017
Partners(s)	Your institution
Tester(s)	Your name
Tester aspect	[Programmer/ Educator / Learner]

Test environment	
Test environment	Give some details about Operating System, Web browser
Test devices	Details about device, e.g. pc, mac, linux

References	
Reference	Give a link to web if applicable

Scenarios				
No.	Testing component and scenario	Expected behaviour e.g. be able to set main menu image	Results round 1	Results round 2
1	More details about what it should do.	Instructions to perform the action and expected result	Ok or not ok	

Issues raised and screenshots

Issue No.	[The unique issue number] e.g. 1
Scenario ID	[Low / Medium / High]
Severity	[Low / Medium / High]
Type	[Bug / Change request]
Summary	[One line summary of the issue]
Description and screenshots	[Description of the issue, please give enough information to reproduce the issue] [Give a screenshot if applicable]
Workaround	[If there is a workaround that mitigates the issue then give it here]
Recommendations	[Recommendation regarding this issue]

In the following we describe for each test, the scenario to follow and the expected behavior.

2. Testing and validating

The testing and validation procedure consists of 5 Scenarios which are briefly

- **Scenario 0**: **Installation** of the platform components by the **Programmer**
- **Scenario 1**: **Content creation** for the **Advanced Educator** and the **Educator**
- **Scenario 2**: **Content creation advanced** for the **Advanced Educator**
- **Scenario 3**: **Editing and deleting** for the **Advanced Educator**
- **Scenario 4**: **Compiling and disseminating** for **both Educator types**
- **Scenario 5**: **Playing** the game for the **learner**

Scenario 0: Installation

This scenario is for the site administrator which should install all the system components in a server. The third party components that should be installed are the following:

1. Unity3D installation at Ubuntu or Windows based server systems: Unity3D can be downloaded for free from Unity3D official site[2]. The version tested is 2017.1. Installation and user activation (for free license) is required.
2. Server software installation at Ubuntu or Windows. Apache and MySQL server can be downloaded in a single package for free from the Apache site[3]. Installation requires the basics for web sites configuration.
3. WordPress installation: WordPress can be downloaded for free from the official WordPress site[4]. Also the theme Evolve should be installed and activated within the WordPress mechanism (Appearance, theme, add new, type evolve, install, activate).

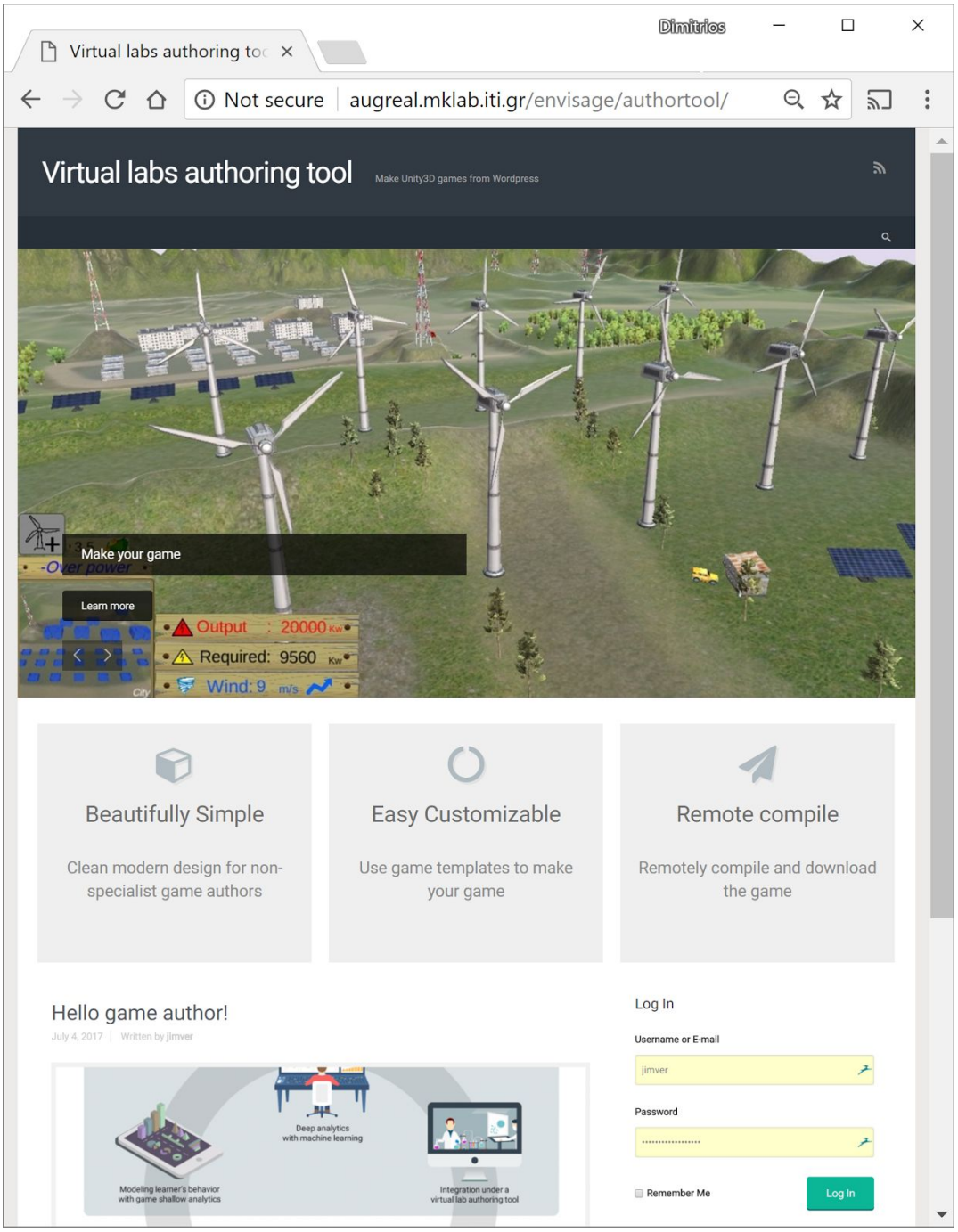
Authoring tool plugin installation

1. The lastest version of the WordpressUnity3DEditor should be downloaded from the

[2] http://Unity3D.com
[3] https://www.apachefriends.org
[4] https://wordpress.com

GitHub repository[5]. Download the repository as a zip.

2. Go to the WordPress site back-end, plugins, add new, and select the zip file for upload.
3. Upon successful installation the back-end menu should have the items Games, Scenes, Assets.
4. Go to the front-end and check if the site appears as in Figure S0.1. The images, the menu, and the login widgets should be configured with the WordPress mechanisms as prefered.
5. Type http://yourdomain/wpunity-main to check the game design procedure. The response should be alike Figure S0.2.



[5] https://github.com/Envisage-H2020/Virtual-Labs-Authoring-tool

Figure S0.1: The Virtual labs authoring tool portal

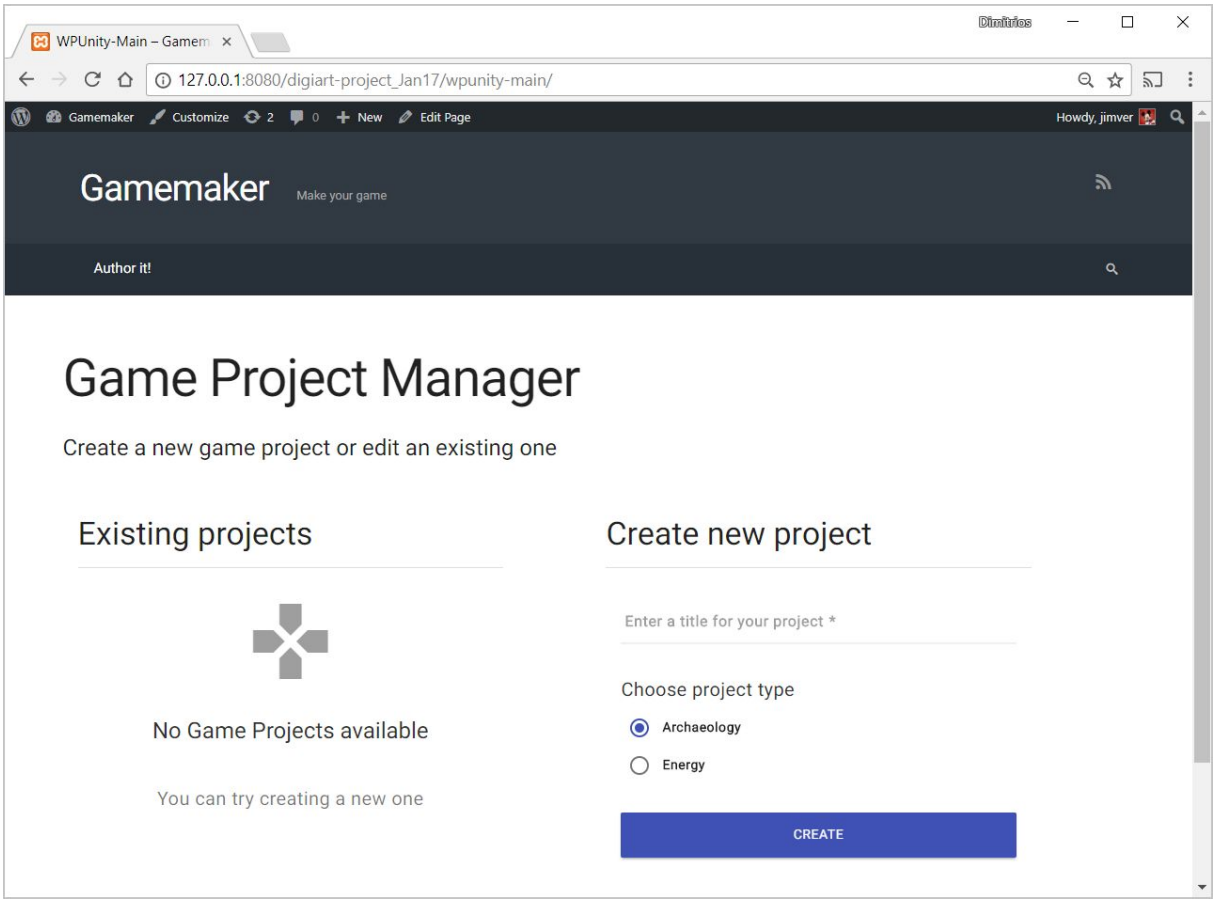


Figure S0.2: The starting page for designing a game.

Testing game analytics: There are currently different options for testing, in particular when using virtual labs in a browser. When integrating the GIO tracking in HTML- and JavaScript-based labs, the preferred integration is based on the Google Tag Manager (GTM). In this case, the tracking can be validated in two different ways as we are explaining in the following.

Test via Google Analytics: When integrating the GIO tracking in a lab based on HTML/JavaScript, GIO's GTM tag is used to forward data from the lab to GIO's servers. We strongly recommend not to only add the GIO tag but also the Google Analytics (GA) tag to GTM configuration. This has the advantage that simple analytics is available out of the box and the data flow can be validated immediately with the help of GA's real-time view. After adding the GTM script to the lab's source code and adding both tags in the GTM console, one can verify that data is being tracked by opening GA's real-time view. When events are now being fired from the virtual lab, these can be immediately seen in GA (see Figure S0.3). However, one should note that GA is not entirely real-time. Active users in the virtual lab are only shown within a 30 minute time window and the available analytics data is very limited. To get full user statistics, one has to wait for the next day.

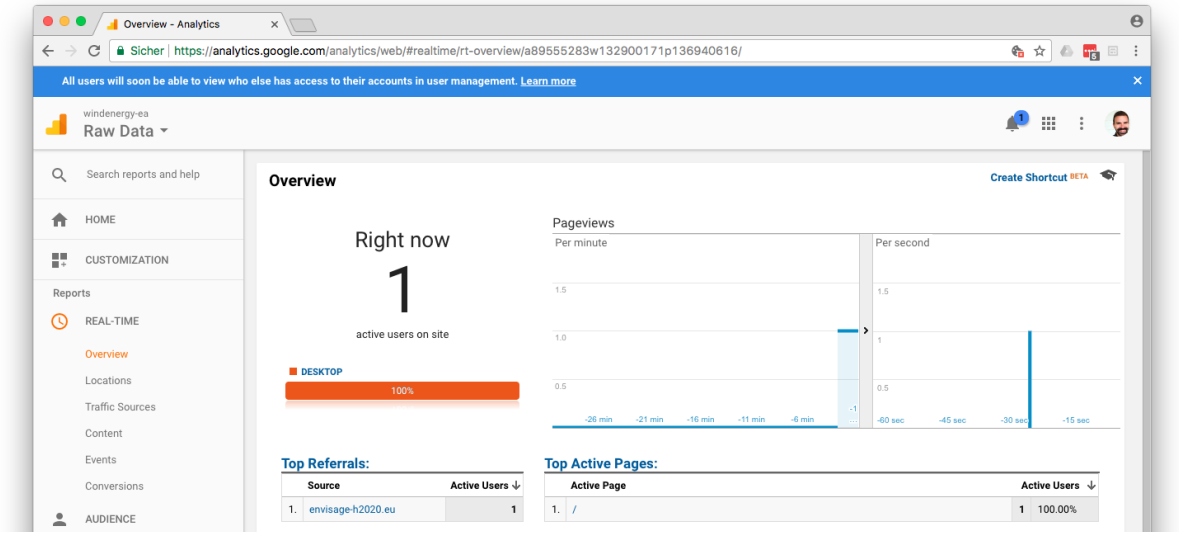


Figure S0.3: Google Analytics real-time view.

Test via GIO Server Response: As described in the paragraph above, GA can be used to validate that data is being tracked and forwarded. However, this does not yet validate that the GIO GTM tag is set up properly and that data is received by GIO's servers. To verify that GIO's servers are receiving data, a more tech heavy approach can be used. As we assume that the integration is being done by a programmer anyways, this test should be easy enough. Modern web browsers, such as Chrome or Firefox, provide tools that show all outgoing requests made by the browser. When GIO's GTM tag is enabled, every trackingpoint will make a request to GIO's server at the address stream.goedle.io. As an example, Chrome's DevTools Console is depicted in the Figure S0.4. It shows a number of request being made to GIO's servers. Each request responding with a status code of 200 is a successful tracked event.

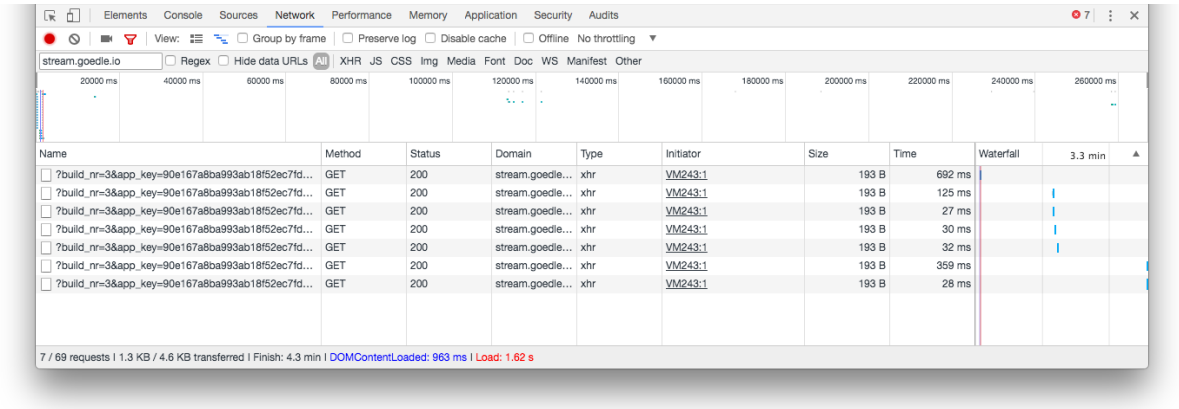


Figure S0.4: Chrome's DevTools Console shows successful data transmission to GIO's servers.

Test Unity SDK: The integration based on the Unity SDK does currently not support a real-time validation. Instead, a test requires up to 24 hours latency in the worst case. After the integration has been set up and tracking points have been fired, the tracked events can be downloaded from GIO's server via the data access API on the next day. We are working on shortening the import cycles and we are assessing different options to provide a simple

real-time validation for the installation procedure akin to GA. However, as our service is not a self-service at the moment, our support team is always cooperative in testing an integration on the fly by checking GIO's backend directly.

Test Analytics Integration: Analytics integration testing can be accomplished immediately using two different methods. The first option is to simply inspect the analytics page in the authoring tool. Given that the analytics tool is represented in an iFrame in the authoring tool, the page loading and rendering as expected, indicates that the analytics server is up and serving pages.

The second option is to request a special URL from the analytics server:

`http://[analytics server location]/healthcheck`

The analytics server will poll each framework used and reply with an html page indicating an OK status for each for these of these if they are available and functioning correctly.

Given that the analytics server implementation only accepts connections from approved locations (i.e. the authoring tool server(s)), these checks can be performed in real-time and with unlimited frequency.

Finally, a summary health check can be completed by requesting:

`http://[analytics server location]/alive`

which returns the string "OK" if the analytics server appears to be running as expected.

Additionally, manual inspection of the server's functionality and ability to render visualizations can be accomplished by visiting the following URL:

`http://[analytics server location]/staticdatacheck`

This URL provides a graphical rendering of a mock dataset collected to allow for manual testing of basic analytics functionality and visual rendering. The page displays the analytics interface in an otherwise blank HTML page, similar to what is delivered to the authoring tools server iFrame.

Scenario 1: Content creation

Content creation is the procedure of generating a Game with Scenes and Assets. First a Game Project should be created. A Game Project consists by default of 3 scenes, namely Main Menu, Credits and First Scene, where the latter is the playable scene of the game. The educator should then upload some Assets that will be available for all the playable scenes of the game. The third step is to insert these assets in the scenes. These steps are explained below.

STEP 1: Create a Game Project

The first step of making a game is to make a Game Project. Upon login, the interface presented in Figure S1.1 is automatically presented.

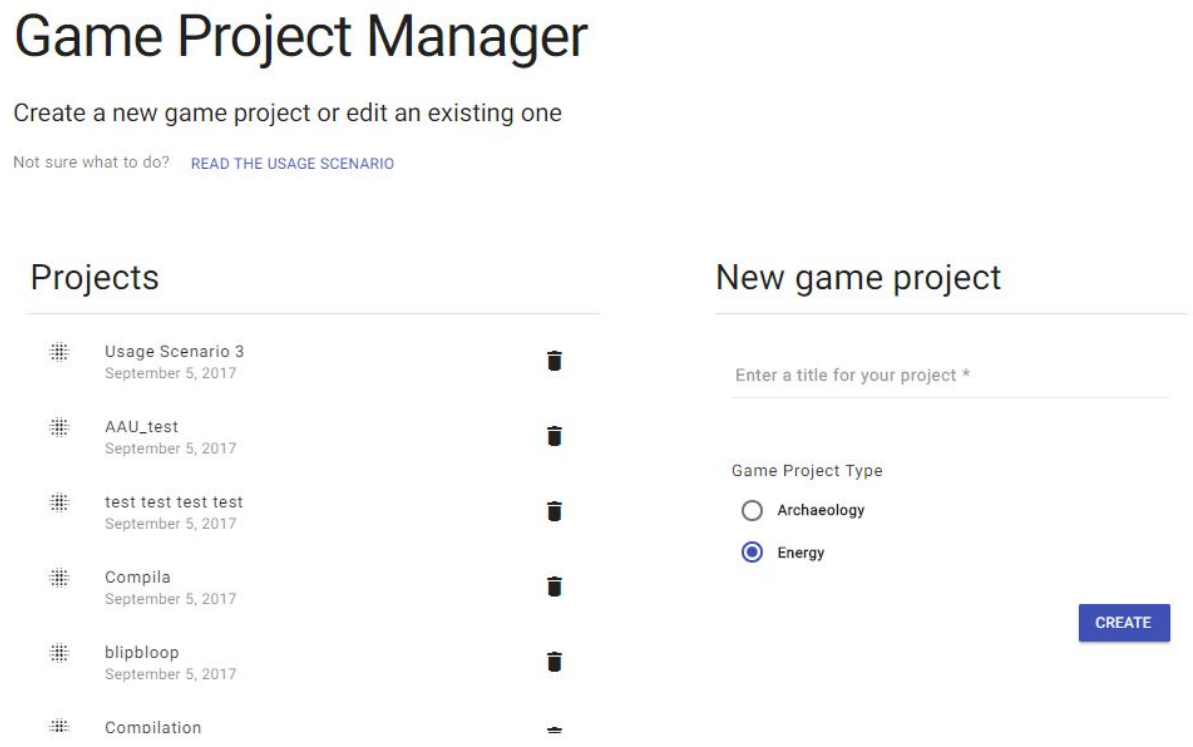


Figure S1.1: The Game project manager allows to create or edit a project.

The Game Project Manager is the central interface for creating a project or editing/deleting an existing one. A new Game Project can be created with the interface shown in the right part “New game project” by entering a title and a type for the project such as “Energy”. In the future more game types will be supported. The game projects can be deleted or edited with the interface on the left by pressing the trash bucket for deleting or pressing the project name for editing.

After creating a game project, Figure S1.2 appears that presents the default scenes of the project such as Main Menu, Credits, and First Scene with tiles. The functionalities allowed in this page is

a) Create an Asset by "**ADD NEW 3D ASSET**" link, which leads to a web page for creating new assets that will be available for all the scenes;
b) Create new scene by **"ADD NEW SCENE"** which leads to the interface for adding second, third and so on playable scenes;
c) Tiles each one representing a scene where the user can edit with "**EDIT**" link or delete with "**DELETE**" link a scene. The default scenes are not allowed to be deleted as the game will not be able to run.
d) Compile the game with the **"COMPILE GAME"** button that will lead to an interface which will be described last.

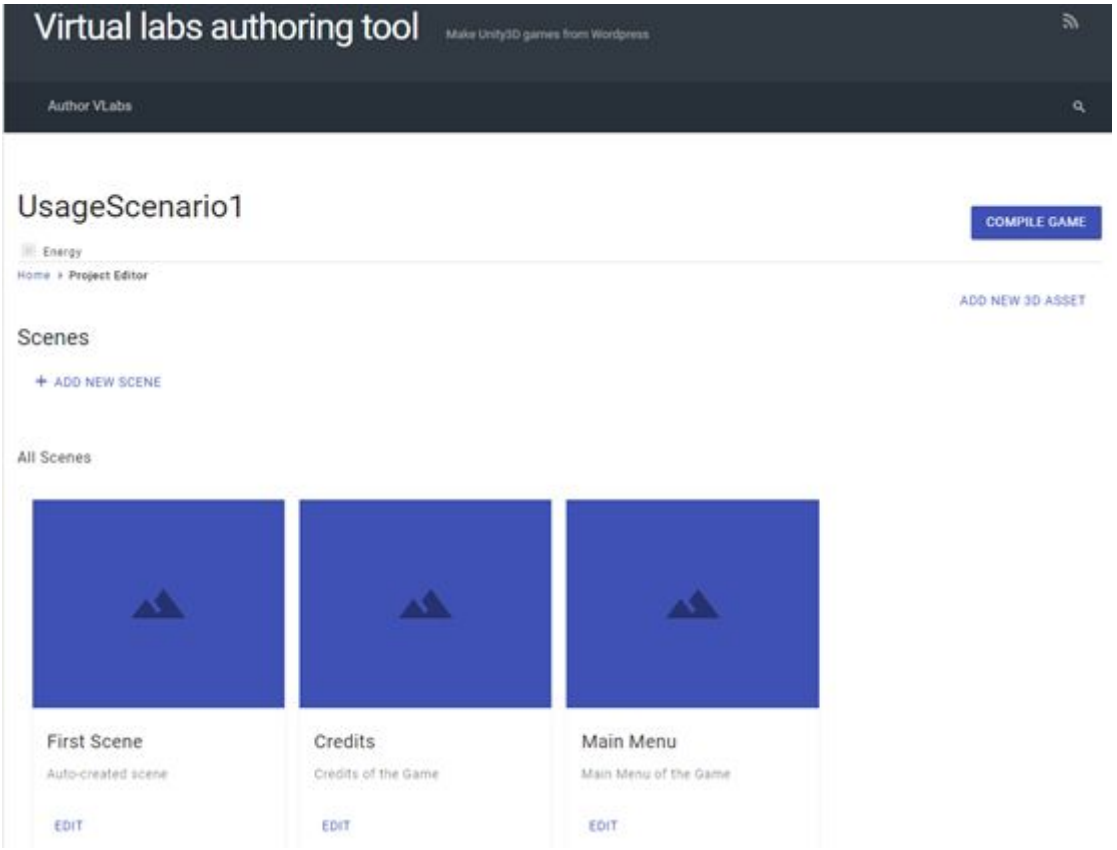


Figure S1.2: Game project editor offers the functionalities of editing a game project.

Step 2: Create Assets

What we need for our first simple virtual lab are three assets, a terrain, a wind turbine that produces energy and a building that consumes the produced energy. The latter two are placed on the terrain. In the project editor as shown in Figure S1.2, press "Add new 3D Asset" button. Then perform the following:

Add a terrain: In the Asset Creator page that will be loaded as shown in Figure S1.3, select in the category dropdown widget the option "Terrain". Several fields will popup. Every Asset should have a 3D model that it is its representation in the game. The 3D model format supported is the Wavefront Obj format that consists of an mtl (material) file that contains color and texture information, an obj file that consists the coordinates of the geometric shape of the model, and a jpg file that consist of the texture of the model. Uploading the mtl, obj, and the jpg for texture. If you don't have any 3D model files, you can obtain various

models for uploading at the bottom of the page where a link to download a starter kit as a zip file is located. Go to the 3D model preview window, zoom in-out with the mouse and press “Create a screenshot” to make an icon for your asset. Then scroll down to see more fields for the Terrain as shown in Figure S1.4. These fields were explained in Scenario 2. After setting all the parameters press “Create Asset” to insert your data in the database

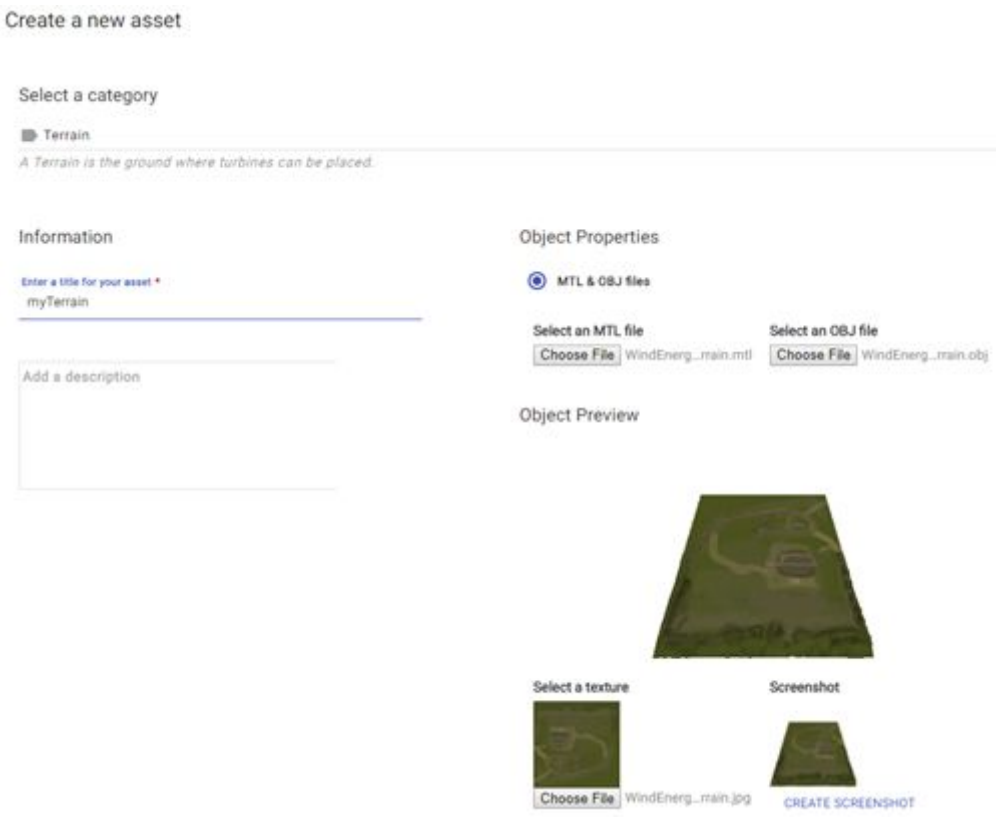


Figure S1.3: Asset Creator page allows to create an Asset where the fields vary depending on the asset category selected.

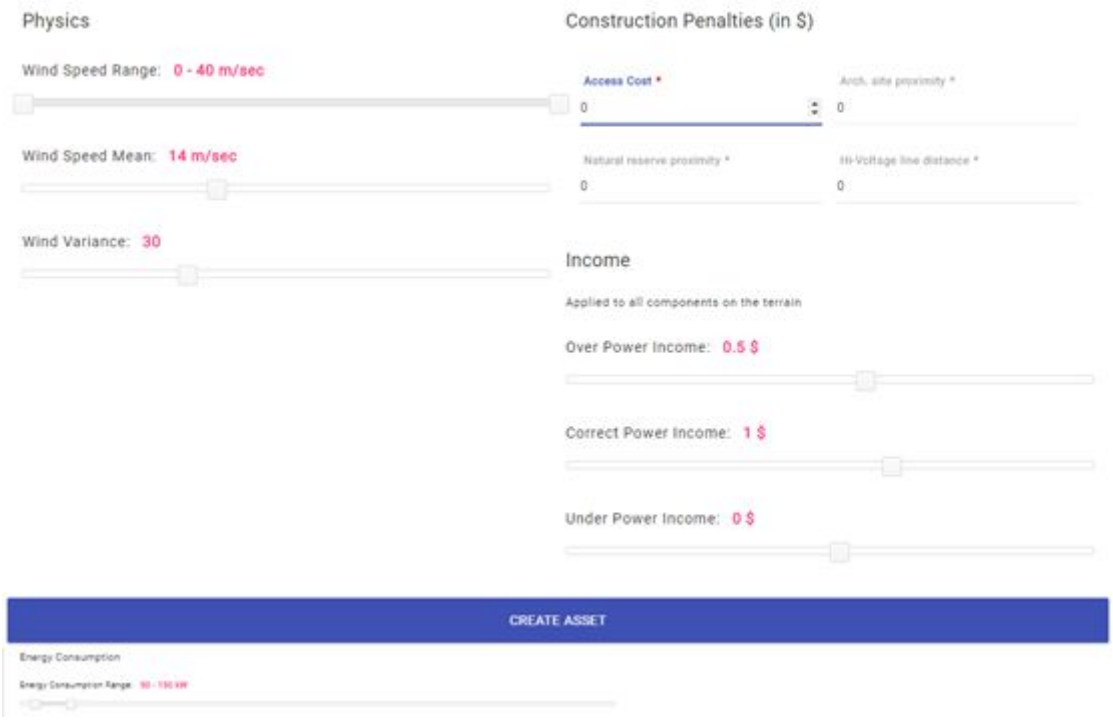


Figure S1.4: Figure S1.3 continued. Several parameters can be selected for the Terrain.

Create a building: In order to add a building, navigate with breadcrumbs to the "Project Editor" and press "ADD NEW ASSET". In the following form as shown in Figure S1.5, select the category of the asset will define which fields should be shown next.

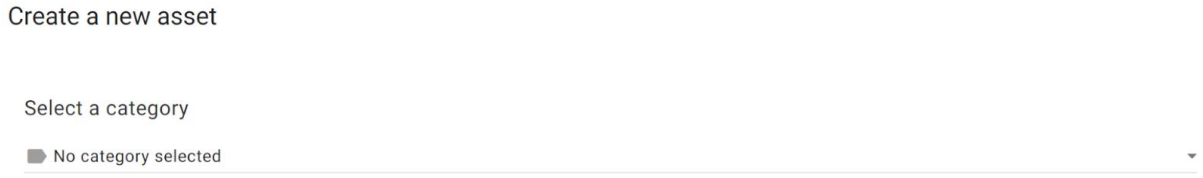


Figure S1.5: Selecting a category for the asset.

For the building, you need to choose "Consumer" category as the building is a consumer of energy. Then, the form will expand showing more fields as shown in Figure S1.6. The fields that should be filled are a) the title of the asset, and b) the 3D files of the asset. Select an mtl and an obj file for the turbine and an image specified for texture. You can locate some 3D models at the bottom of the form as link to download resources. Rotate the presented 3D model in a good looking pose. Press create screenshot to generate an icon for the asset for the selected pose. Leave the building's properties, i.e., the energy consumption range, mean and variance, to the default values. These parameters will be described in Scenario 2. Press "Create Asset" in the end of the page.

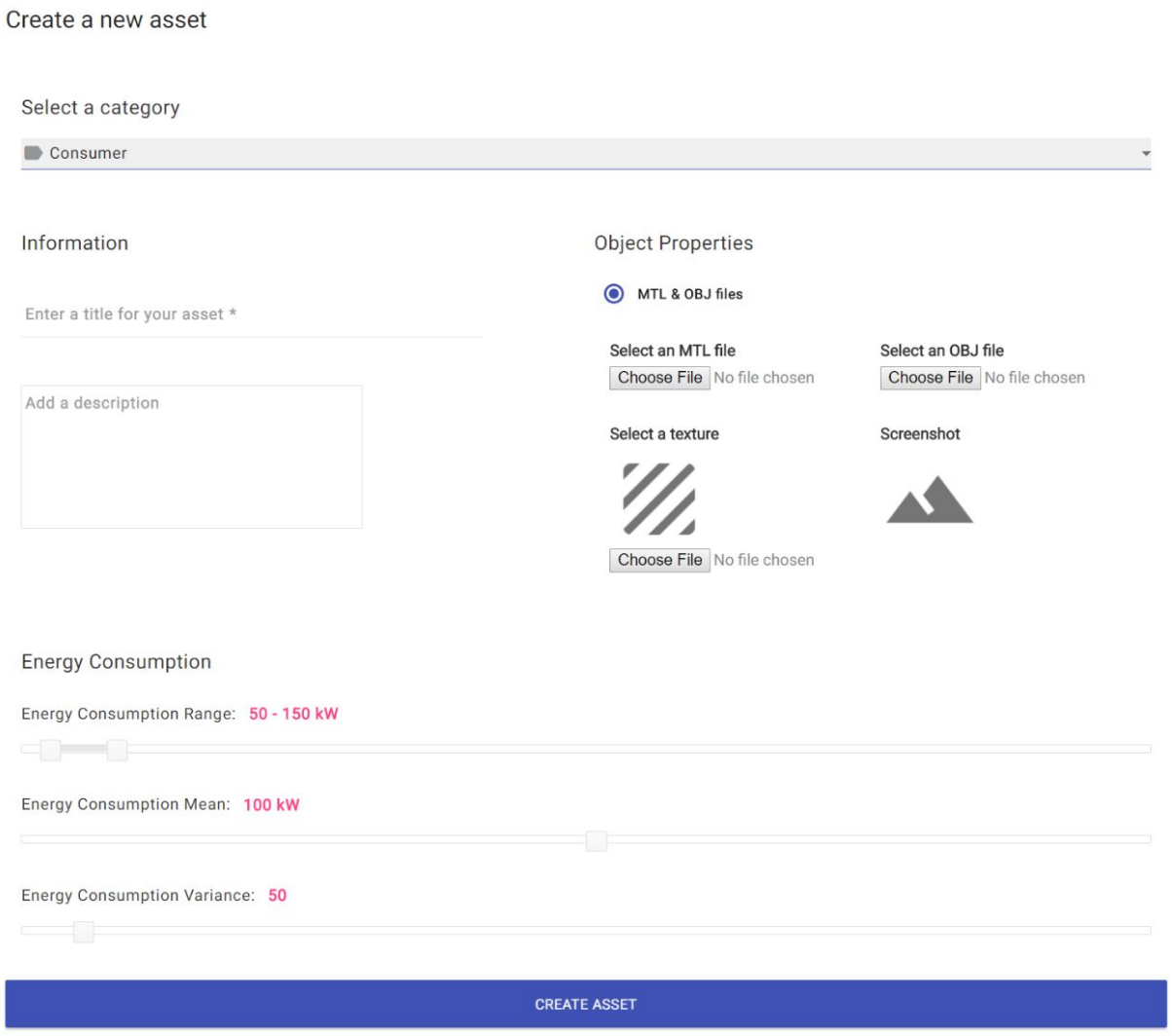


Figure S1.6: Form for creating a building.

Create a turbine: In order to add a turbine, navigate with breadcrumbs to the "Project Editor" and press "ADD NEW ASSET". In the following form as shown in Figure S1.7, select the category of the asset will define which fields should be shown next. For the turbine, you need to choose "Producer" category as the turbine is a producer of energy.

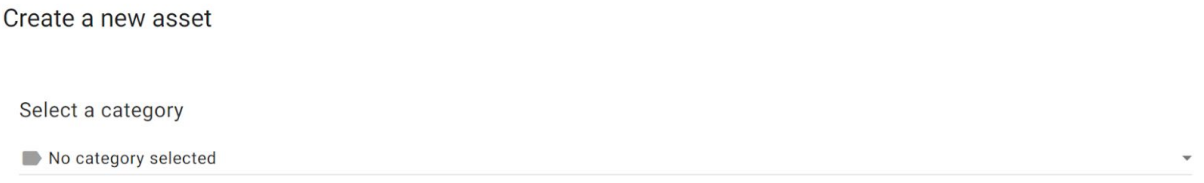


Figure S1.7: Selecting the category of the asset.

The form will expand showing more fields as shown in Figure S1.8. The fields that should be filled are a) the title of the asset, and b) the 3D files of the asset. Select an mtl and an obj file for the turbine and an image specified for texture. You can locate some 3D models at the bottom of the form as link to download resources. Rotate the presented 3D model in a good looking pose. Press create screenshot to generate an icon for the asset for the selected pose.

Leave the turbine's properties, i.e., the production efficiency depending on the wind speed, as it is. More details on these fields will be found in Scenario 2.

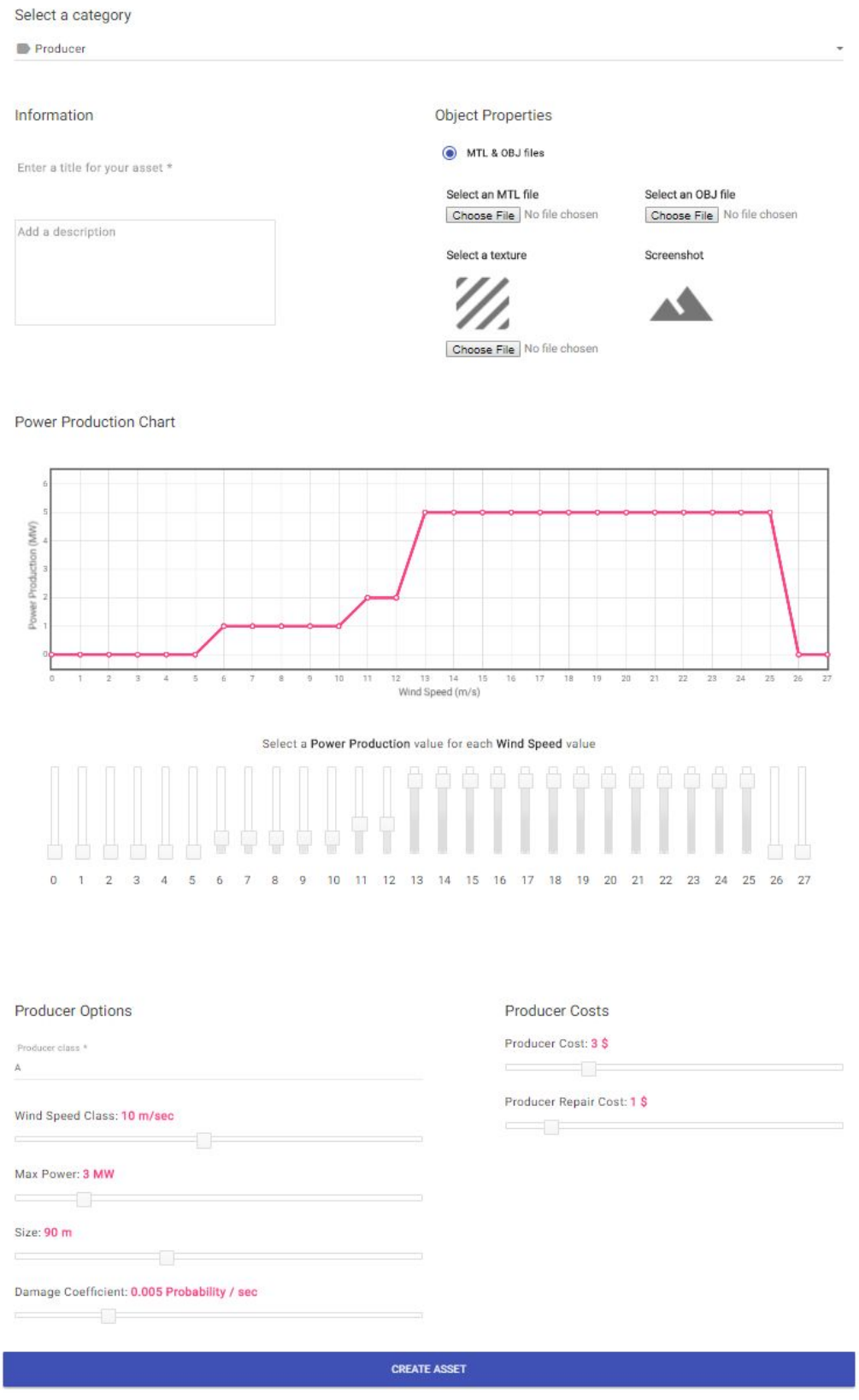


Figure S1.8: Creating a turbine asset.

Step 3: Add assets to the scene

The assets should be placed inside a scene so that it is working properly. The minimum requirement for the scene to be functional is that a terrain should be placed. Go to the Project Editor from breadcrumps as shown in Figure S1.9. Initially, three “scenes” are available by default i) a 3D scene where the game takes place, ii) the main menu and iii) the credits. Click on the “edit” button of the first scene in order to edit it. A 3D editor will popup as the one presented in Figure S1.10.

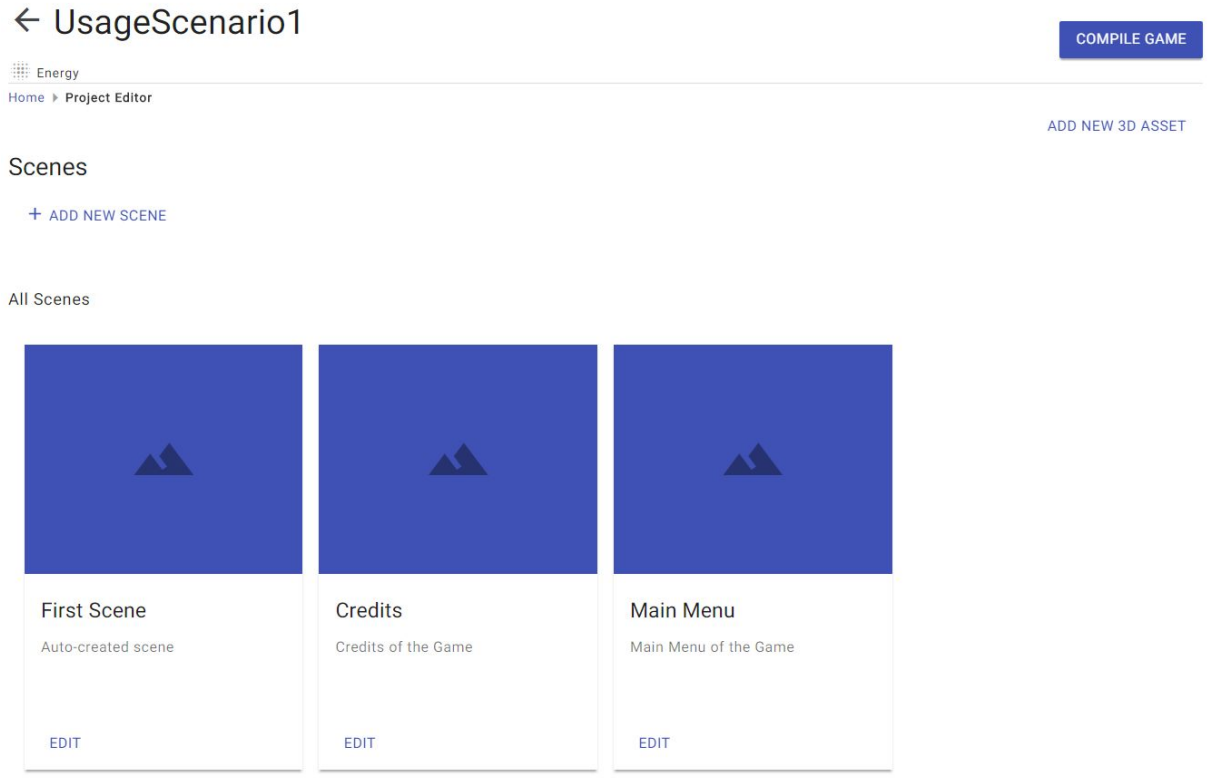


Figure S1.9: Project editor contains the scenes to be edited

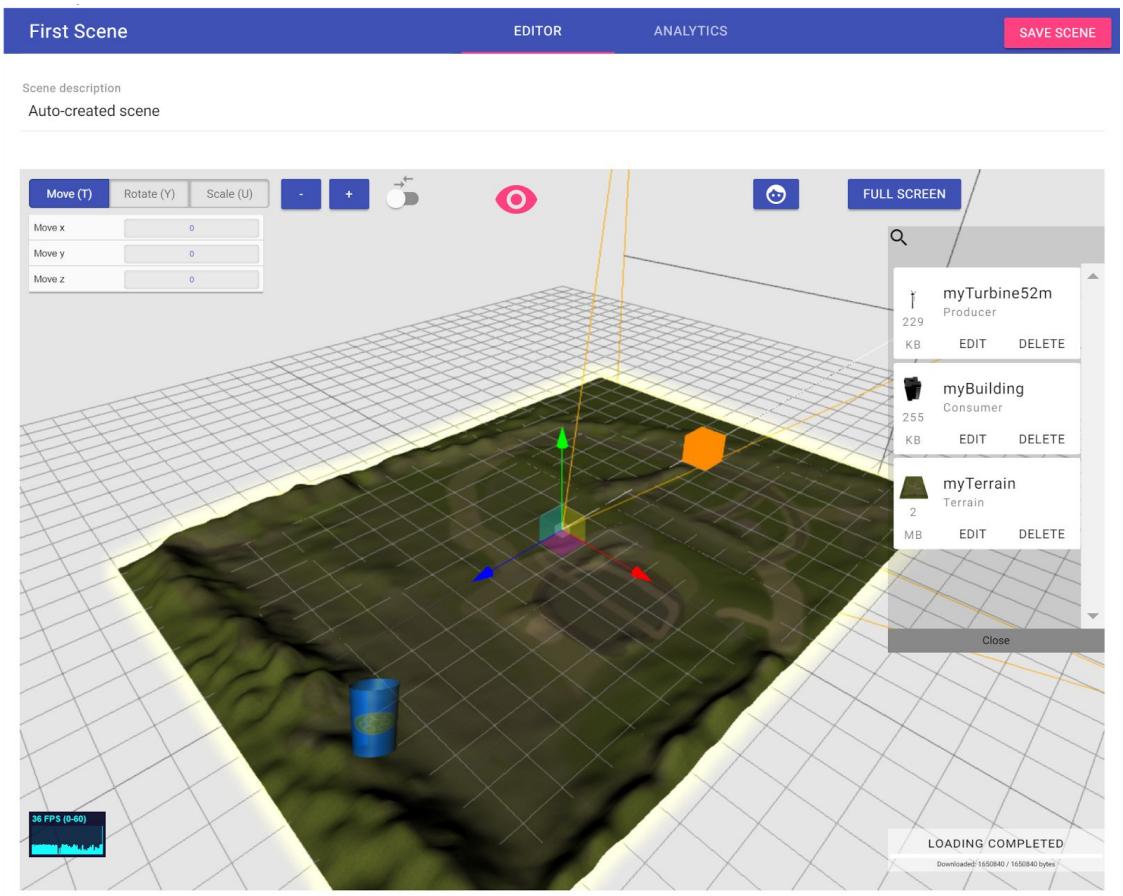


Figure S1.10: The 3D editor allows for editing a scene.

The 3D editor consists of the following components

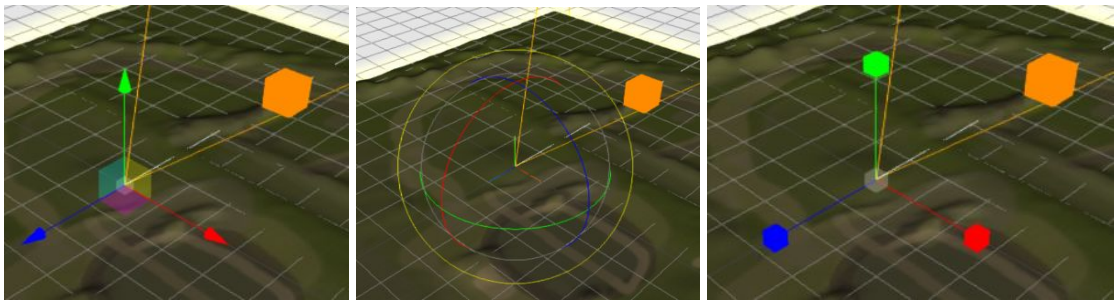
a) The upper ribbon which contains the title of the scene which can be clicked and edited, the "EDITOR" button which is by default selected and shows the 3D environment of the scene, the "Analytics" button which shows the game analytics, and the save button that saves all the changes in the scene. Below the ribbon, the description of the scene can be found and edited.

b) The 3D environment which can be used to edit the scene. The 3D environment contains the avatar, the right sidebar of available assets, several buttons, and the recycle blue bin.

   i) The avatar is the physical representation of the player. The avatar position and orientation is stored and inherited in the compiled game as the initial player position and orientation. The avatar is also the orbital point in the scene. This means that by holding LMB (left mouse button) and moving the mouse you will orbit around the avatar and see whatever is nearby him. You can zoom in/out around the avatar with the scroll wheel. You can move the avatar with RMB (right mouse button) and moving the mouse. The avatar orientation can be changed by pressing the button with the face  on the upper right which shifts the orbital view to first person view and by moving the mouse.

   ii) The sidebar of assets can be used for inserting assets into the scene. An asset can be inserted multiple times in the scene. This is called instantiation of an asset as a game object. All game objects inherit the properties and the behavior of the asset. The sidebar can be moved with LMB to other places

and it can also be closed to save working space. In order to insert an asset into the scene, drag-n-drop the asset tile into the 3D environment. The asset is placed at the avatar position and a 3-axes widget with arrows is displayed (gizmo) as shown below in Figure S1.11a. By using LMB on the arrows the gameobject can be moved (translated) to another position. This can be also achieved using the "Move" button. In order to rotate the game-object, click with LMB onto the orange cube or press "Rotate" button and the gizmo will be changed into rotation mode as shown in Figure S1.11b. Using LMB click and mouse movement, the game object can be rotated. The game object can be scaled by pressing again the orange cube or the "Scale" button. The gizmos will change as in Figure S1.11c. Using LMB click on the gizmo the object can be scaled. The game object is always scaled uniformly for all axes and only positive scaling is allowed (no inside out scaling). The 3 modes change in a cyclic fashion by pressing the orange cube.



a. Move mode b. Rotation mode c. Scale mode

Figure S1.11: The gizmos of a) movement; b) rotation; and c) scaling.

iii) The recycle bin serves for deleting the game objects from the scene. Drag-n-drop a game object into the recycle bin and it will be placed inside the bin. If you click with LMB on the recycle bin, all the game objects are presented and if you click on a game object then it is placed back into the scene. If you place "Save scene" button the scene is saved without the items in the recycle bin.

In the editor, choose a terrain, a consumer and a producer and place them in the scene: Drag and drop one wind turbine and one building after inserting the terrain, which all have been created as described in the previous section. Adjust the above assets: place and move them accordingly so as they seem naturally placed on a terrain's spot. Determine also their scale. After finishing press "Save scene". Typically, the scene should look like in Figure S1.12.

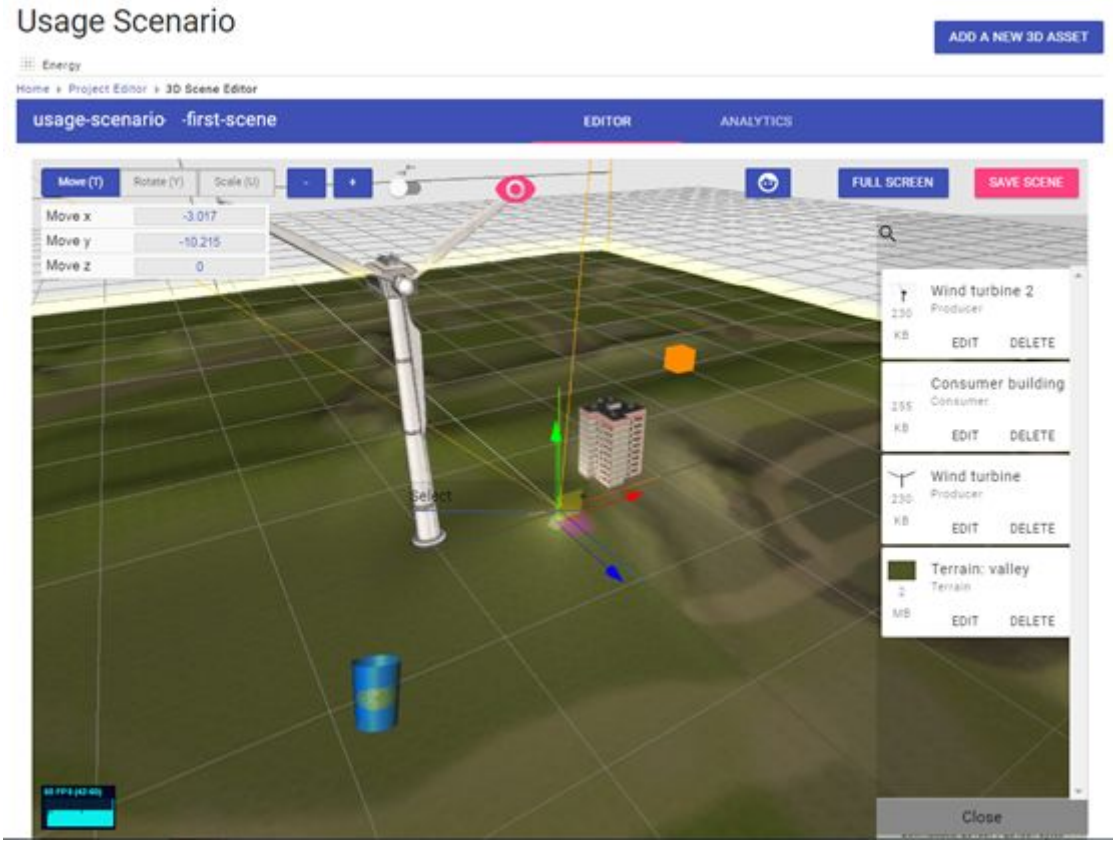


Figure S1.12: A scene with a terrain, a turbine, and a building.

Step 4: Edit the game main menu.

The Main Menu is the first screen of the game where the learner can start interacting with the game. The presentation of the Main Menu in the game is based on a predefined parameterized template which is shown in Figure S1.13. The parameterization allows to change the background image, and hide Help, Login, Options buttons.

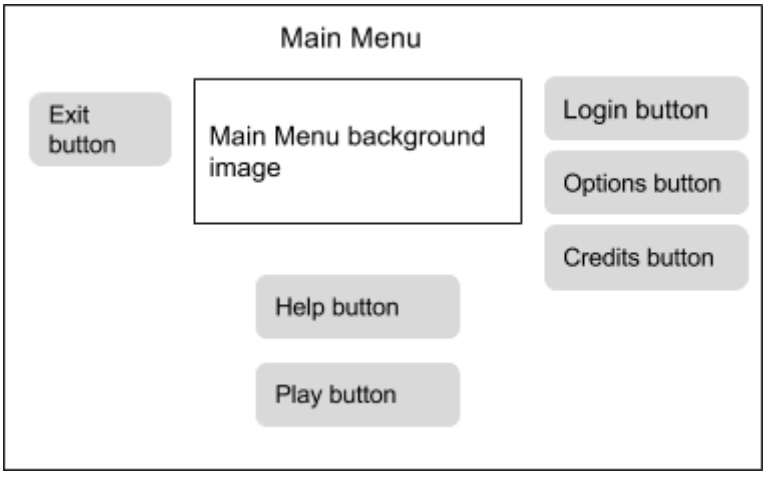


Figure S1.13: Main Menu scene template. See Figure S5.1 for an instantiation.

In order to change these parameters, perform the following actions. Using the breadcrumbs go to the Project Editor, press onto the “EDIT” button of the “Main Menu” scene tile. A form as that of Figure S1.14 will appear.

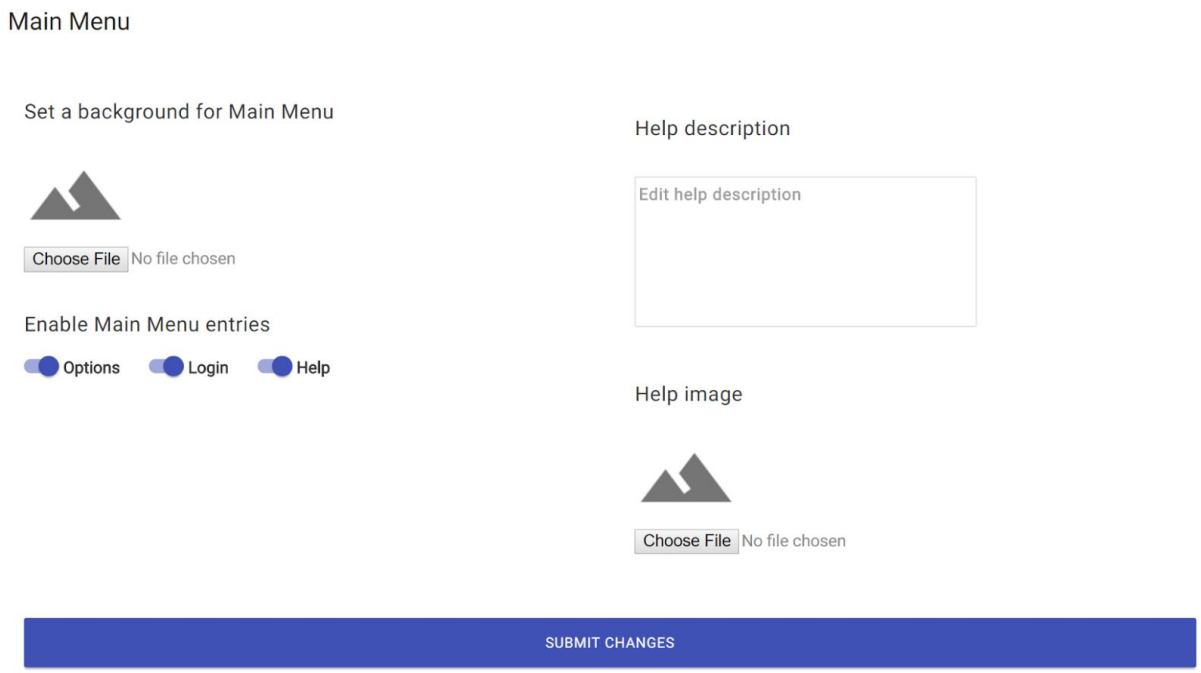


Figure S1.14: Form for editing the “Main Menu”

In this form, select first a background image (jpg is only supported) for the Main Menu. Optionally, the login, options, and help scenes can be deactivated. For the “energy” game leave them in the default state.

In the right part of this form you can modify the Help scene. The Help scene will be presented as an image with a paragraph below the image as shown in Figure S1.15. Then select an image (jpg) for the Help scene and a text for the Help scene.

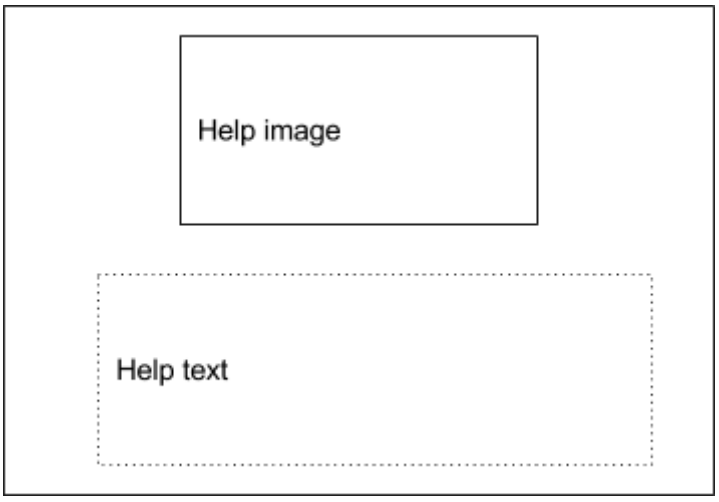


Figure S1.15: Help scene template. See Figure S5.2 for an instantiation.

Step 5: Write the credits

Credits are about the persons that contributed to the specific project and the creation and design of the virtual lab, as well the design of the 2D and 3D assets. The credits are presented as in the template of Figure S1.16.

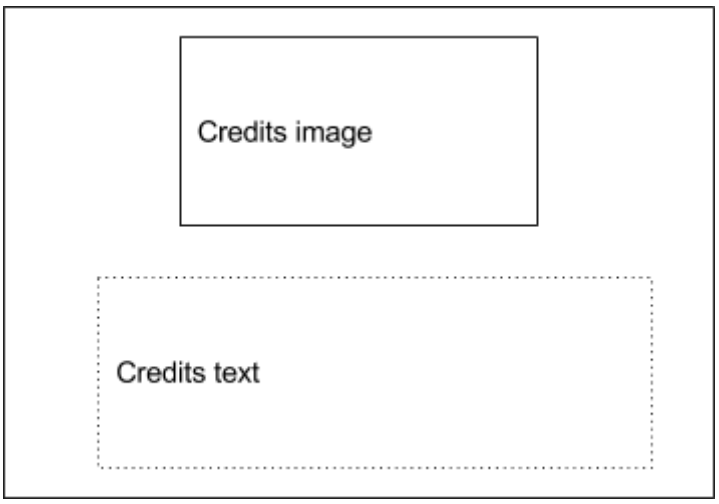


Figure S1.16: Preview of the credits scene in the game. See Figure S5.4 for an instantiation.

The Credits scene can be modified as follows. Select “Project editor” from the breadcrumbs, and select “EDIT” in the Credits tile. Then, the form of Figure S1.17 will appear. Select an image for the background and write down text for the credits.

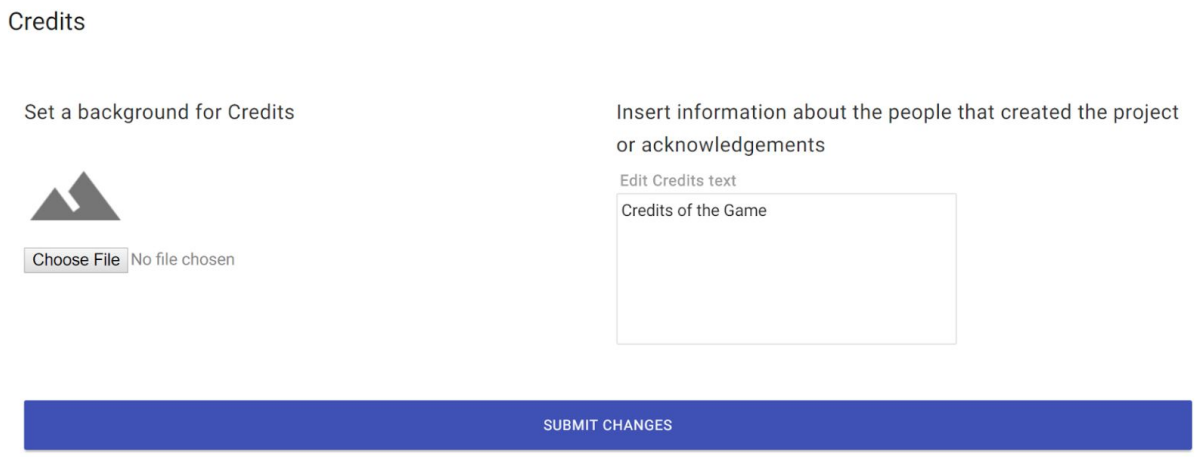


Figure S1.17: Selecting the image and text for the credits scene.

Scenario 2: Content creation advanced

This usage scenario aims to demonstrate the functionalities offered by the authoring tool in a more advanced way, and particularly in focusing on the wind terrain, turbines, and building properties. It is assumed that Scenario 1 was executed and basic knowledge of the authoring tool is obtained.

Terrain parameters: After login, select an available Game Project or create a new one. Then press the button “ADD NEW 3D ASSET”. Select in the category dropdown widget the value “Terrain”. Then several fields will popup. We will skip the explanation of the fields of 3D models as they were described in Scenario 1. The rest of the fields are shown in Figure S2.1.

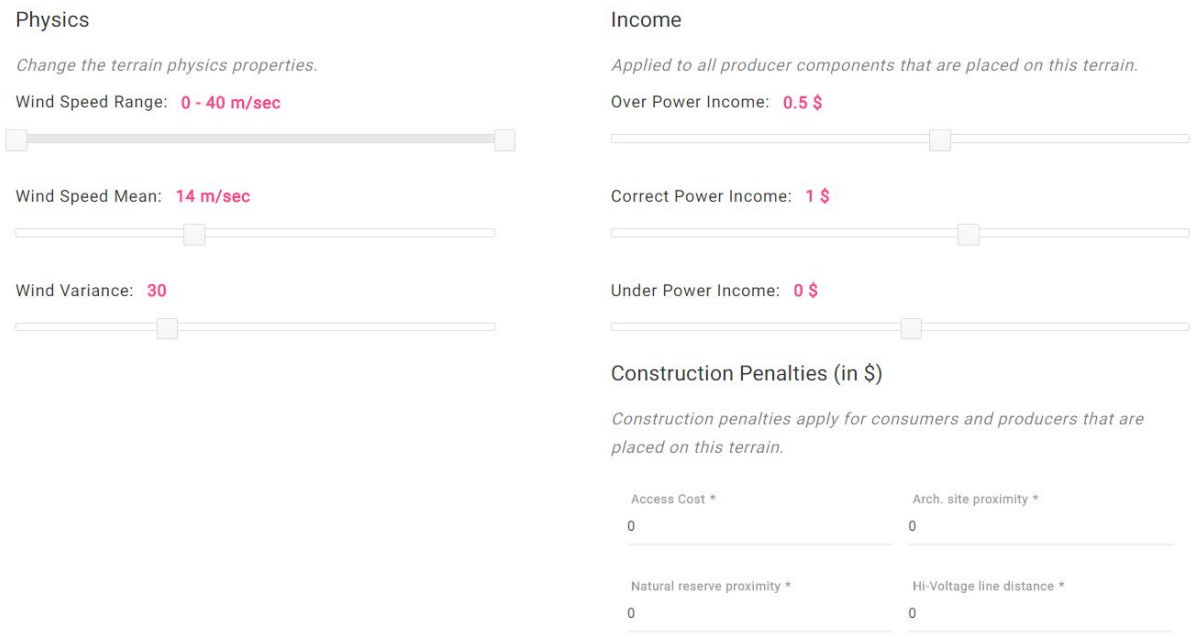


Figure S2.1: Parameters for the Terrain.

Physics parameters contain the wind parameters such as “Wind Speed Range” which limits the wind speed air within the lower and upper limit, “Wind Speed Mean” which defines the average wind speed air given a Gaussian distribution, and “Wind Variance” which defines the variance of the wind given a Gaussian distribution.

Income parameters are related to the virtual money earned when the balance of the energy of the scene is “Over Power” that is the produced energy by turbines is more than the requested by the buildings, “Correct Power” that is the energy produced/required is in balance, and “Under Power” when the requested energy is more than the produced.

Construction penalties parameters contain the virtual money to be subtracted when several factors affect the installation of the wind park in certain areas. Access cost is the penalty for areas that are difficult to access. Archaeological site proximity is the penalty when the wind park is nearby an archaeological site. Natural reserve proximity is the penalty when the wind park is near a natural resource such as national parks. High voltage line distance penalty is

when the wind park is far from high voltage lines. Next we will describe the Turbines parameters

Turbine parameters: Select in the category dropdown widget the value “Producer”. Apart from the 3D model files, several parameters will popup. The first parameters, as shown in Figure S2.2, are related to the production of energy of the wind turbine (Y Axis) on several wind speed values (X Axis). Through the sliders at the bottom the educator can change the response of the turbine for each wind speed.

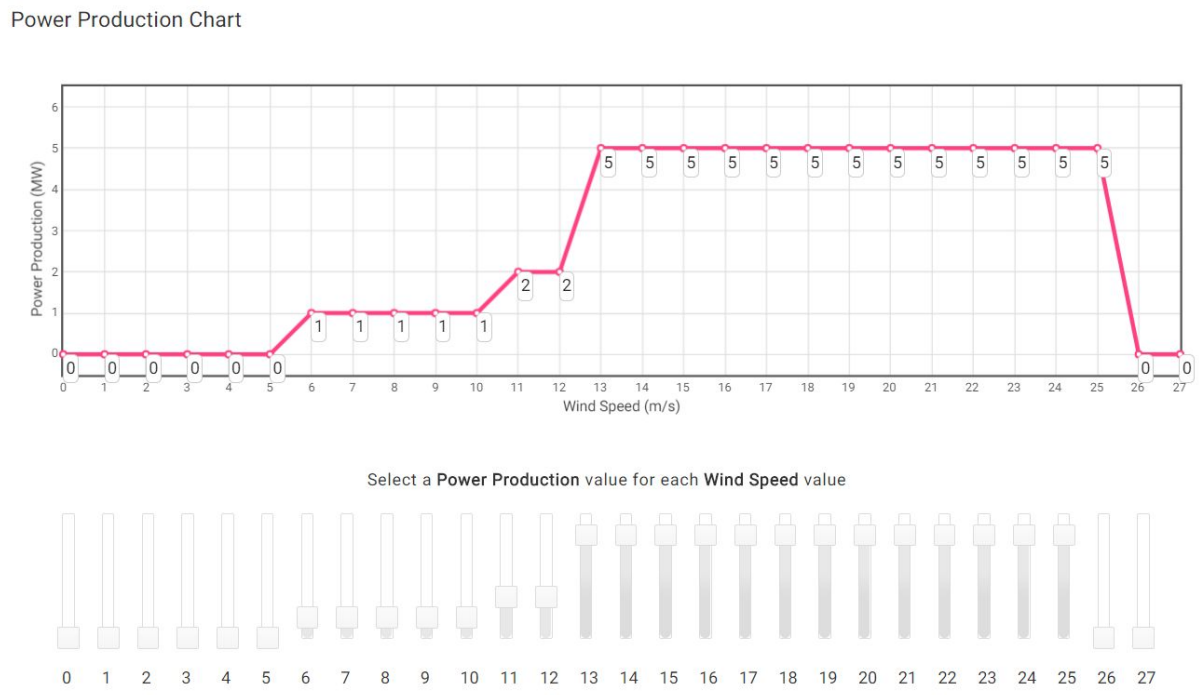


Figure S2.2: Energy production curve of the wind turbine.

By scrolling to the bottom, several other parameters can be found as shown in Figure S2.3.

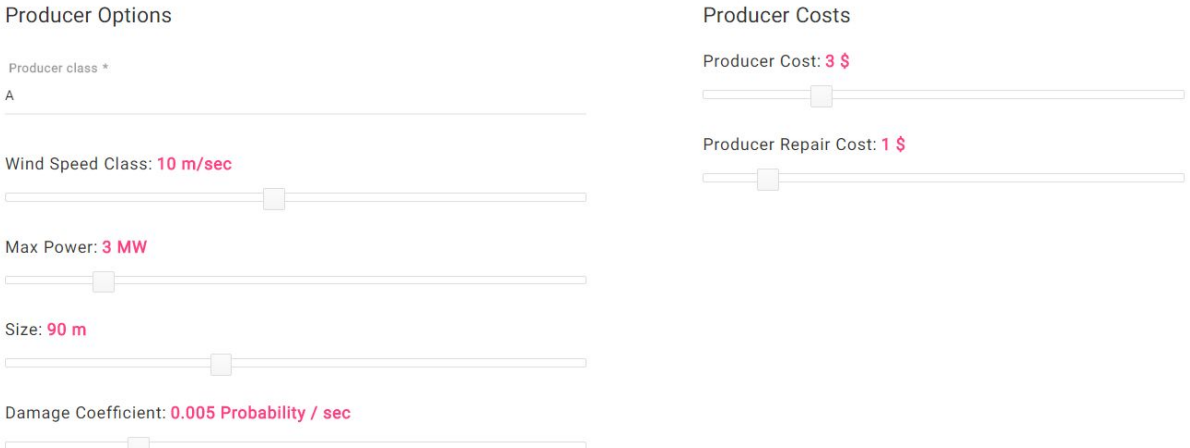


Figure S2.3: Parameters of the wind turbine regarding production and cost.

The parameters Producer Class, Wind Speed Class, and Max Power and textual information for the learner before building the turbine that allows to take a decision accurately. The Size

of the turbine regards the height of the pylon of the turbine which is about the same as the rotor diameter. The size is important for setting various parameters inside the game such as the marker for selecting the turbine and the minimum distance of the nearby turbines. The damage coefficient is the probability for a turbine to break down at a certain timestamp. The producer cost is the money subtracted from the learner's score when the turbine is build. The producer repair cost is the money reduced from the score when a turbine is repaired.

Building parameters: In the category of assets, select "Consumer" which is a term to denote everything that consumes energy, mainly buildings. The parameters that will popup are shown in Figure S2.4. Energy consumption range contains the upper and lower limit of the energy consumed from the building. Energy consumption mean and variance define the the energy consumption as a value from a Gaussian distribution with the respective values.

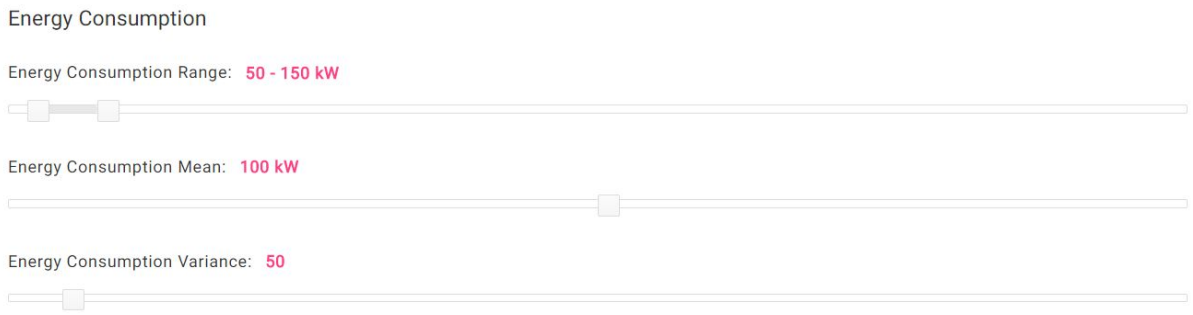


Figure S2.4: Parameters for the buildings

Adjust the above assets. Go to "Project Editor" from breadcrumbs or with the back button. Make a new scene or select the first scene. Drag and drop six wind turbines and six buildings after creating the terrain. A single asset can be dragged multiple times in the scene. Place them accordingly so as they seem naturally placed on a terrain's spot by moving and rotating them. An example is seen in Figure S2.5.

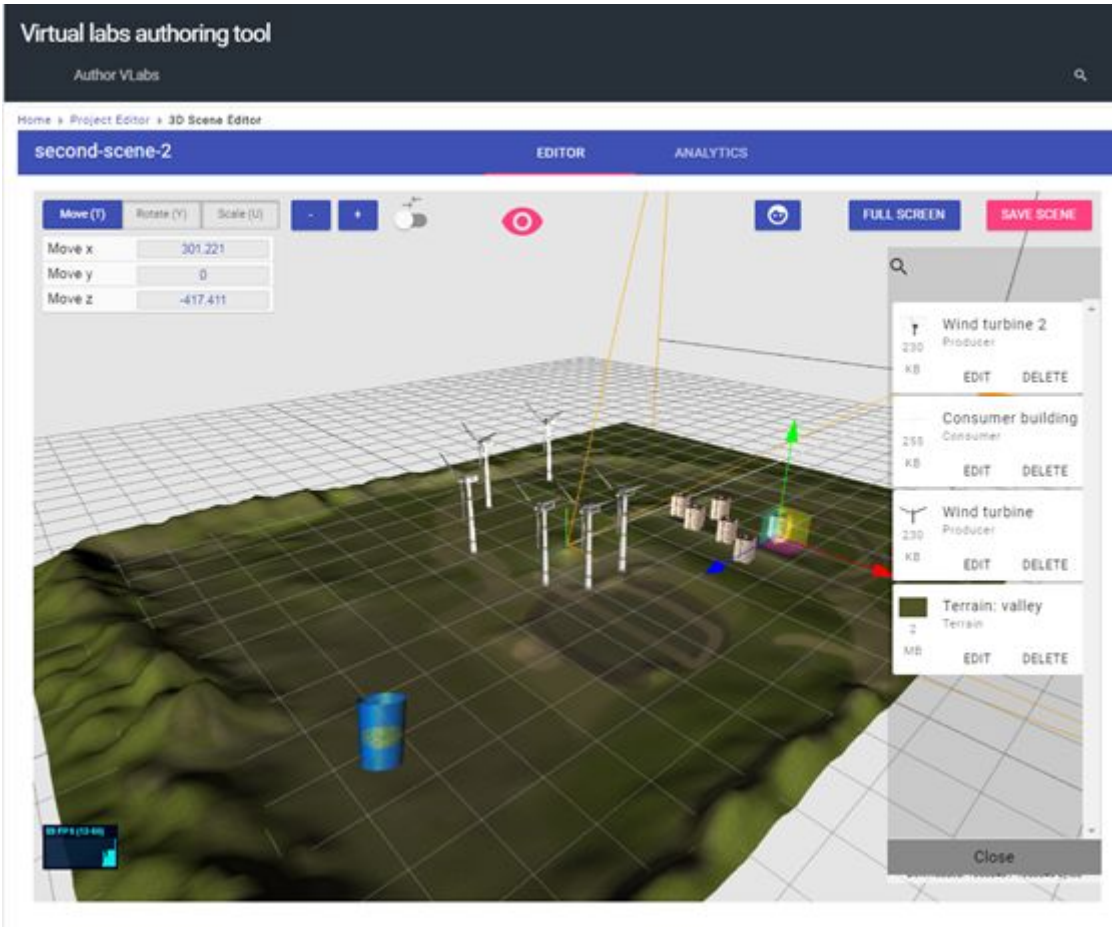


Figure S2.5: An asset can be dragged-n-dropped several times in the scene

Scenario 3: Editing and deleting

Editing and deleting is also supported. The game project can be deleted by pressing the trash bin as shown in Figure S3.1. A dialogue will follow for confirmation. There is no way for restoring the game project after deletion.

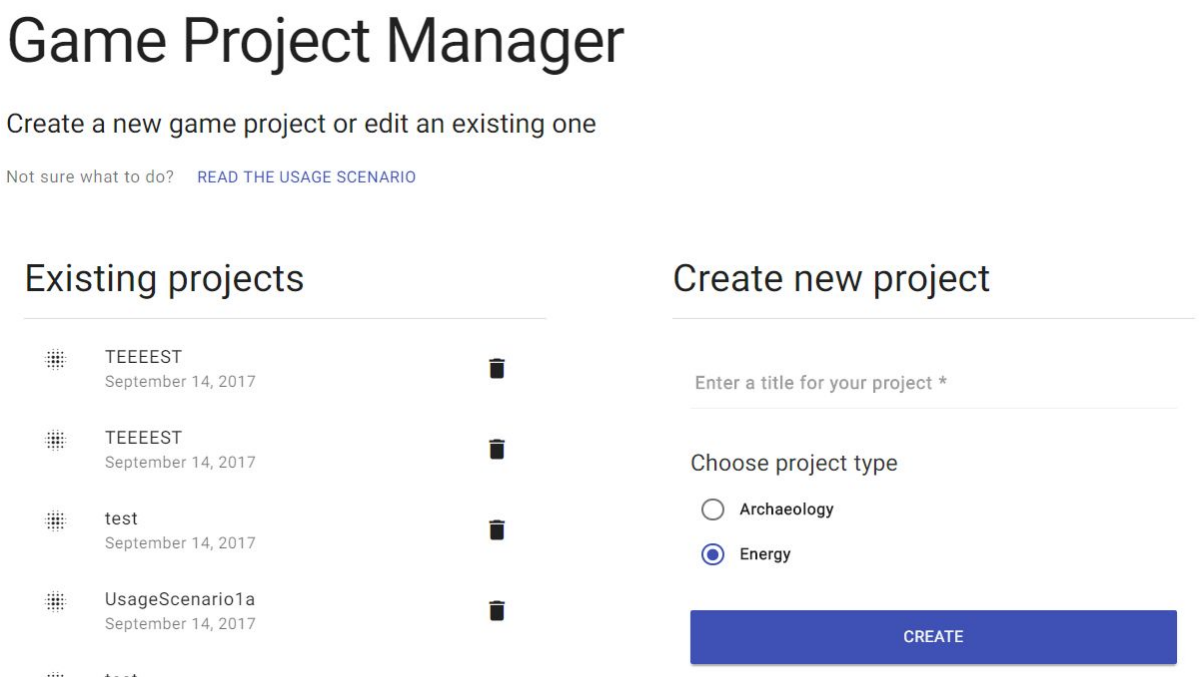


Figure S3.1: A game project can be deleted by pressing on the trash bin nearby.

A scene can be edited or deleted from the Project Editor. Select from Project Manager a Project and the Project editor will popup as shown in Figure S3.2. All scenes can be edited, however the default scenes “Main Menu”, “Credits”, and “First Scene” can not be deleted. A scene can be deleted by pressing the “DELETE” button in the bottom of the respective scene tile. In order to edit the title and the description of the scene press on the “EDIT” button and in the 3D editor edit the title in the ribbon and the description text below the ribbon as shown in Figure S3.3. Then press “SAVE SCENE”. The button will change color twice and the scene will be saved. Reload the page to ensure that it is saved correctly. For editing the content of the 3D scenes or the 2D scenes, go to Scenario 1 where the procedure is explained thoroughly.

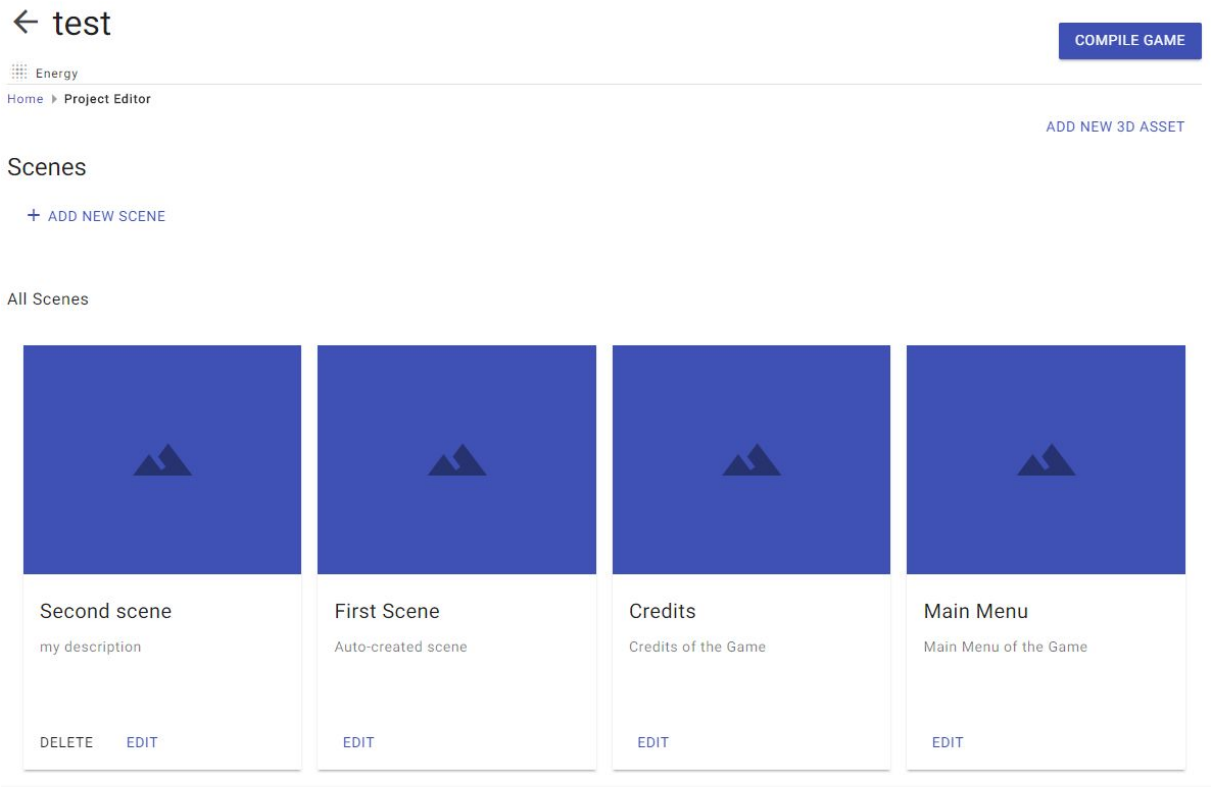


Figure S3.2: Editing and deleting a scene.

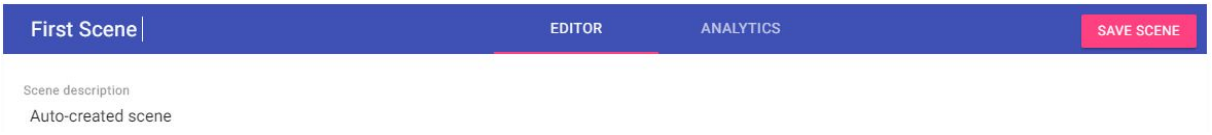


Figure S3.3: Editing the title and the description of the Scene.

An asset can be edited or deleted through the right sidebar of the 3D Editor. In order to go to the 3D editor press on “EDIT” in any 3D scene tile. The 3D editor will be loaded and on the right the list of assets will be loaded, as shown in Figure S3.4. In each asset an “EDIT” and a “DELETE” button can be found, where “EDIT” popups the asset parameters for editing and the “DELETE” button deletes the asset. The 3D files of the asset can not be modified but only its parameters.

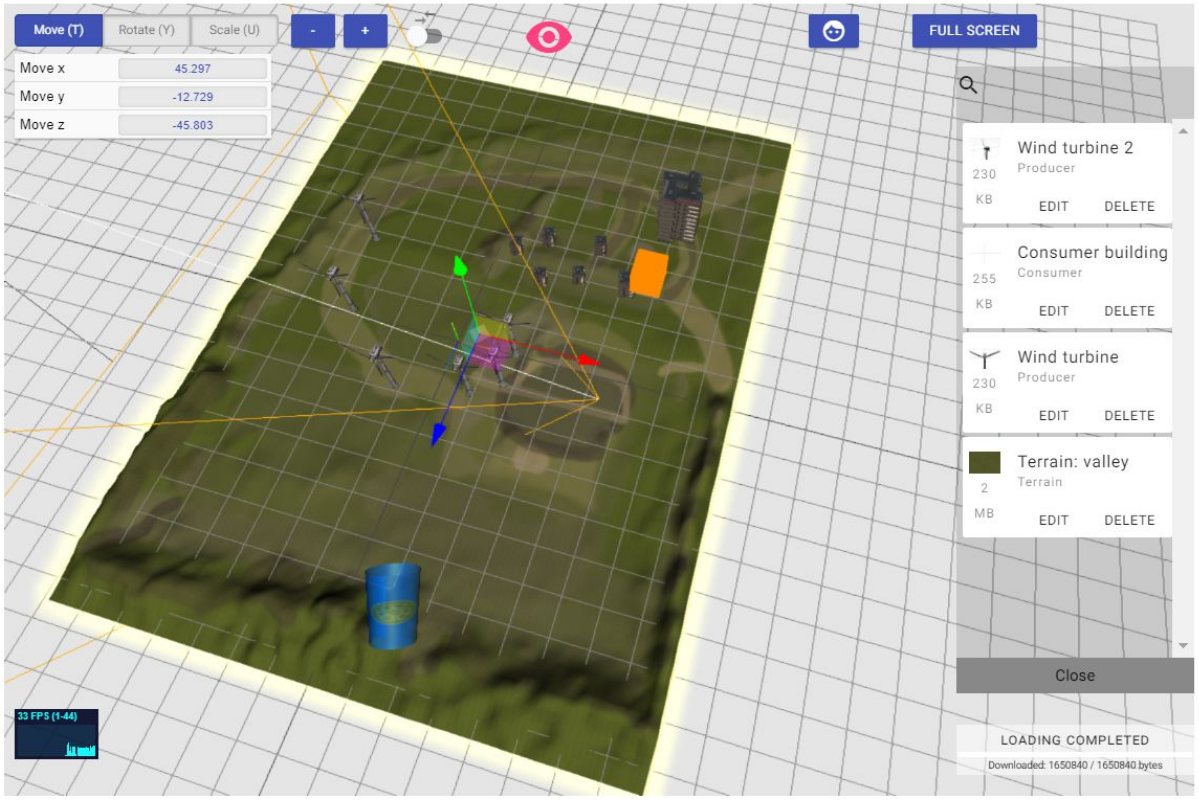


Figure S3.4: Edit and Delete asset can be found in the right sidebar of 3D editor

Scenario 4: Compiling and disseminating

The compiling of the game project towards a binary or a WebGL page is the last step of the game creation. The compiling consists of two steps that are seamlessly executed in the back-end, namely the execution of the compiling, and the zipping and downloading of the binary. If the compiling is for the web, a link for playing the game without downloading is given.

It is assumed that a game was generated, assets were uploaded, and the scenes have assets inside them as it was described in Scenarios 1, 2, and 3. Therefore the game is ready for compiling. The procedure to test the compiling is as follows. Select an already available game from the Project Manager. In the Project Editor page press on the button “COMPILE” on the top right corner as shown in Figure S4.1.

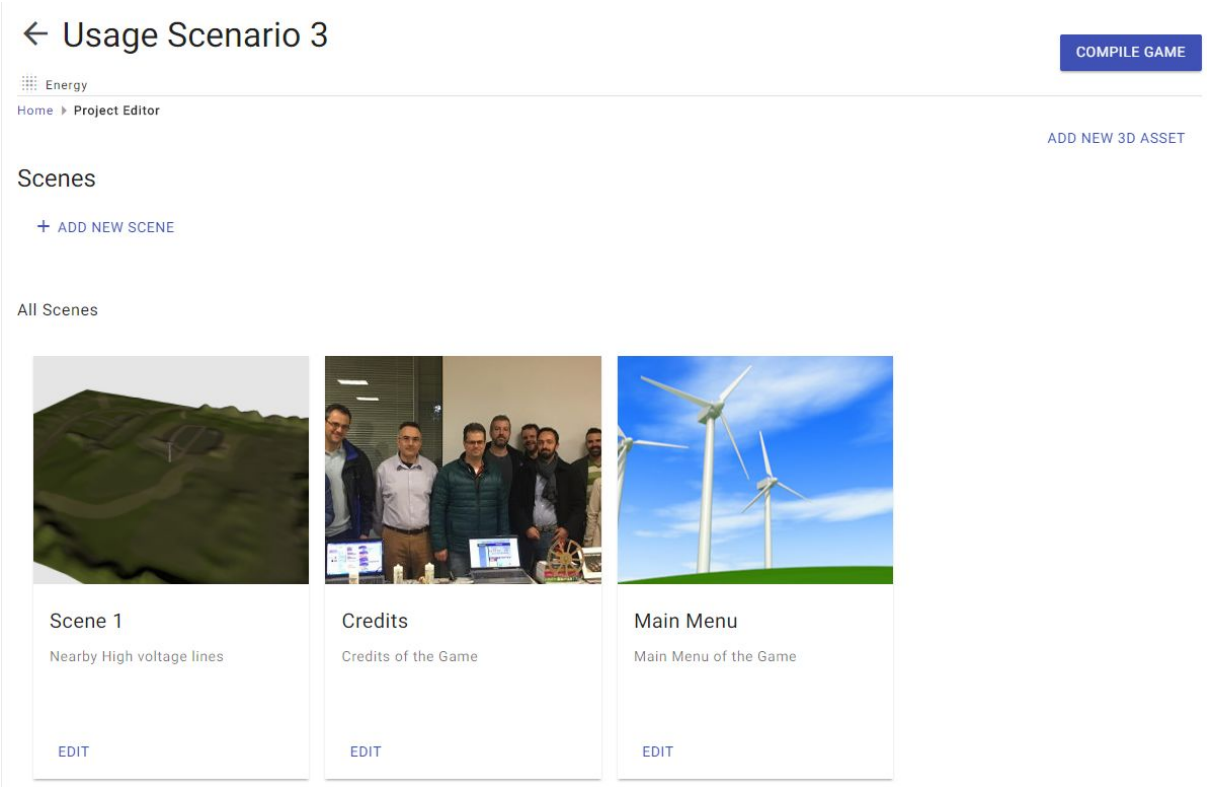


Figure S4.1: “Compile Game” button is on the top right corner of game manager

Then select the platform targeted for compiling by using the drop down menu in the left as shown in Figure S4.2. Available platforms are Web, Windows, Mac, and Linux. In the future Android and other platforms will be supported.

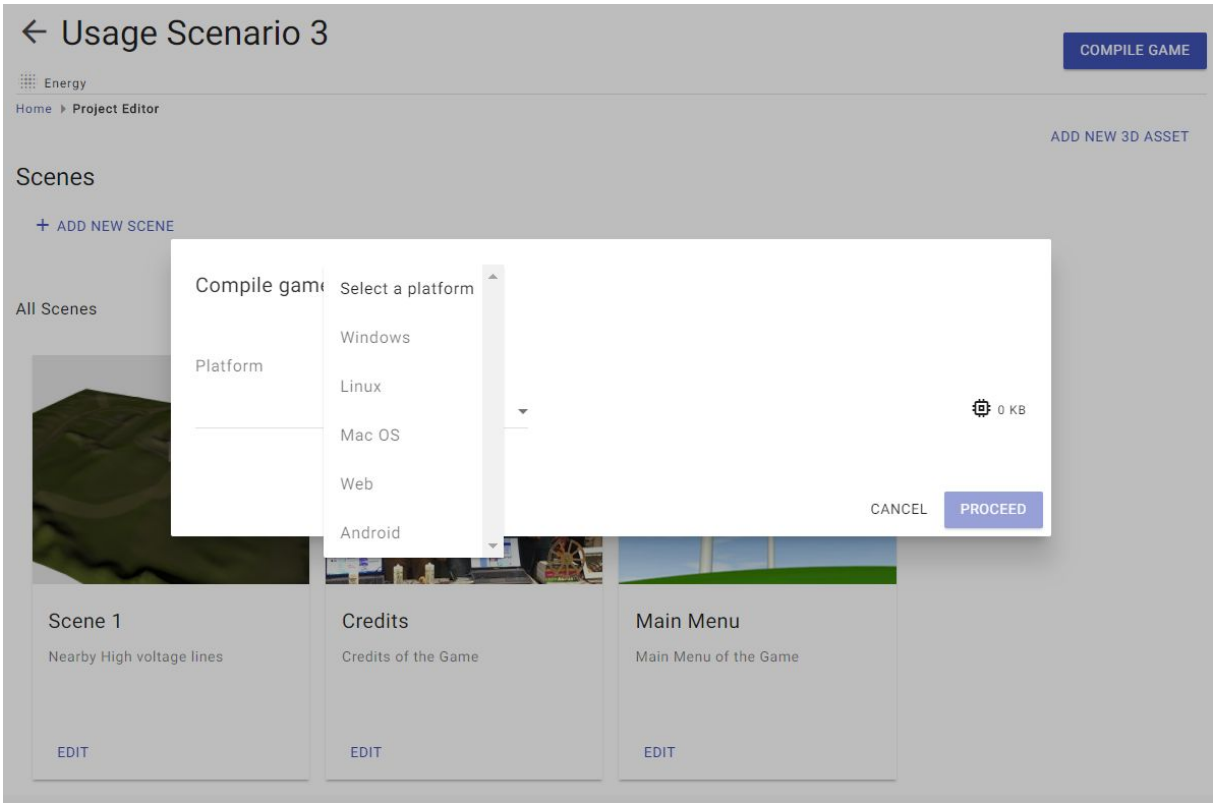


Figure S4.2: Select a platform for compiling the game.

Select Windows for the targeted platform and press “PROCEED”. The compiling process will begin and the interface of Figure S4.3 will appear. The interface consists of a progress bar of 4 steps for Windows. A number at right side indicates the memory of the process at the server which will start from a low value and should be updating through time.

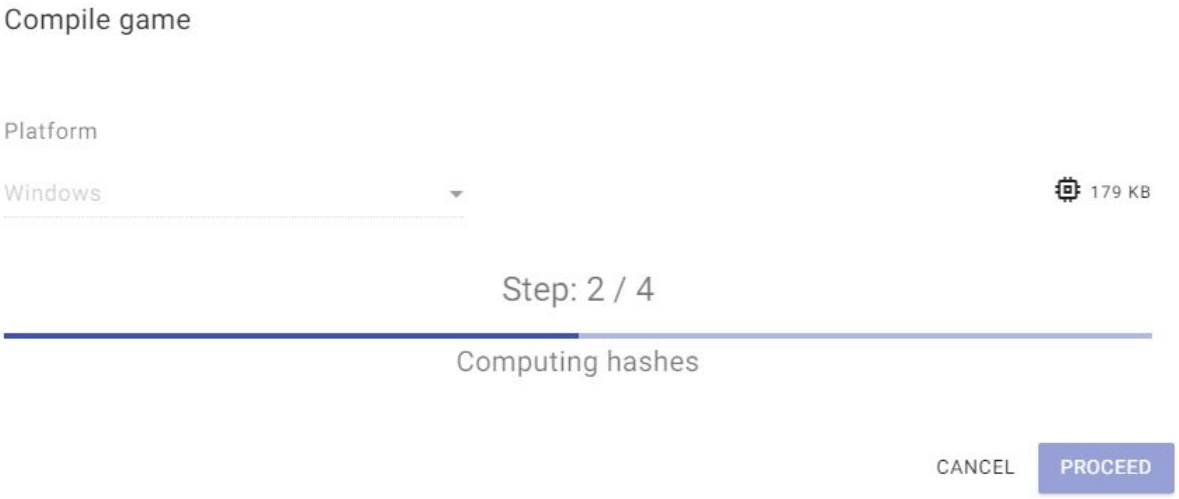


Figure S4.3: Compiling for Windows desktop standalone binary.

If the compiling is successful then the interface of Figure S4.4 will appear. It consists of a link to download the binary in zip format. If the game is compiled again, the new binary will replace the old one. Typical time for a windows compile is 4 minutes depending on the amount of scenes and assets in the scenes.

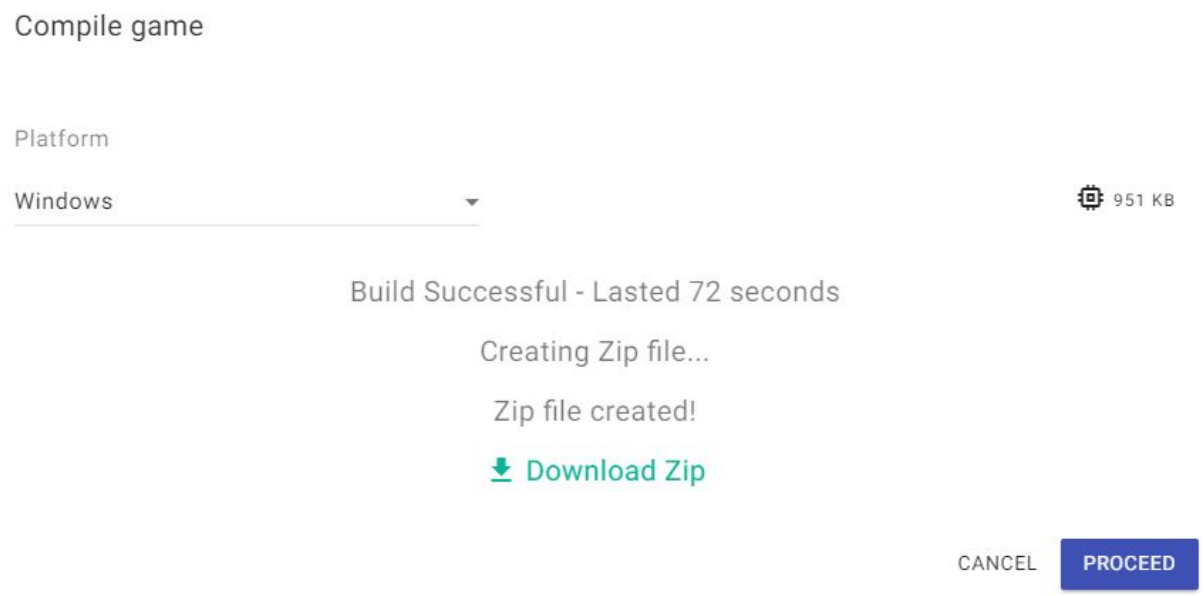


Figure S4.4: Successful compiling for Windows.

The game can be compiled by selecting "Web" in the Platform dropdown widget and by pressing "PROCEED". The compiling for the web lasts much longer than the Windows compile as it consists of 11 steps as shown in Figure S4.5.

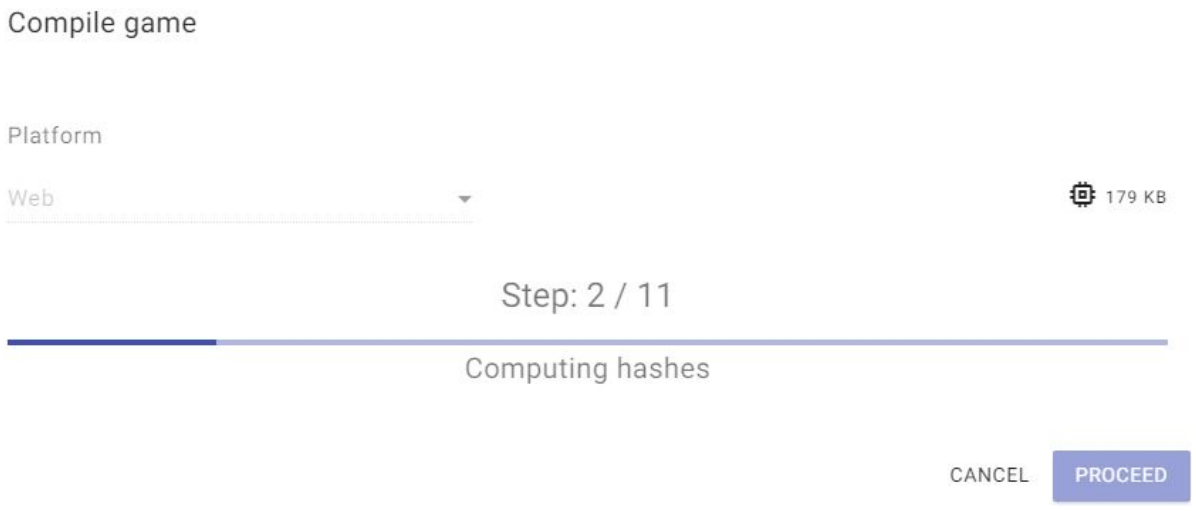


Figure S4.5: Compiling for web

If the game is compiled successfully, then the interface will be that of Figure S4.6, consisting of two links. The first link is to download the HTML code of the game in order to install it on a server, whereas the second link is the the link to play the game on the Authortool server. This link can also be shared with students to play the game. The typical lapsed time for web compiling is 6 to 10 minutes depending on the amount of scenes and assets in the game.

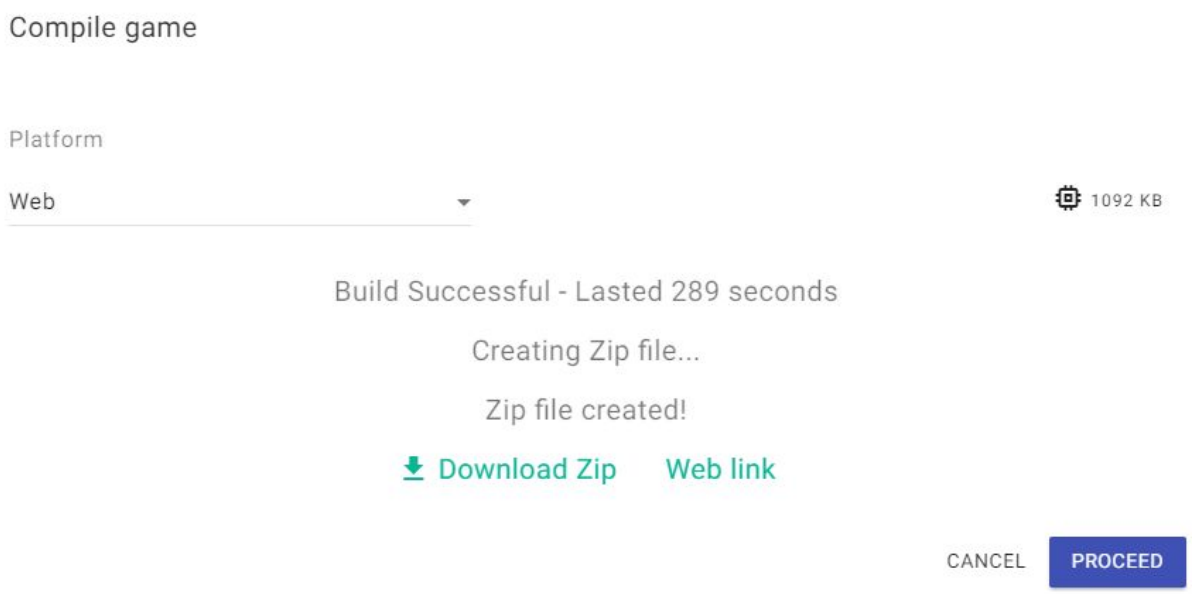


Figure S4.6: Successful compiling of game project into a web page.

Scenario 5. Playing the game

After compiling for web in Scenario 4, press “web link” to play the game. Alternatively, if the game was compiled for Windows, Mac, or Linux, download the zip files by pressing “link to download zip” file, decompress it and press on the “mygame.exe” to play the game. Ignore any certification questions as our server is in testing mode and does not produce digitally signed games. The initial page of the game is the Main Menu as shown in Figure S5.1. Be sure that the image is the one selected in Scenario 1 for Main Menu scene.

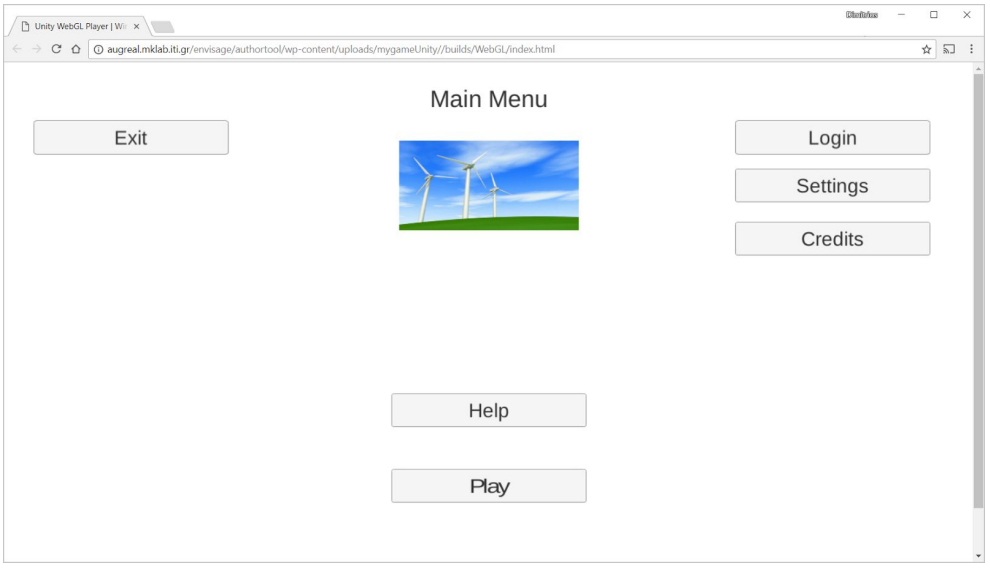


Figure S5.1: Main menu of the compiled game.

Press on “Help” button and ensure that the image and the text are the ones entered in Scenario 1.

Figure S5.2: Help scene of the compiled game

Press on Login and ensure that the following interface is presented. Provide credentials and press back button.

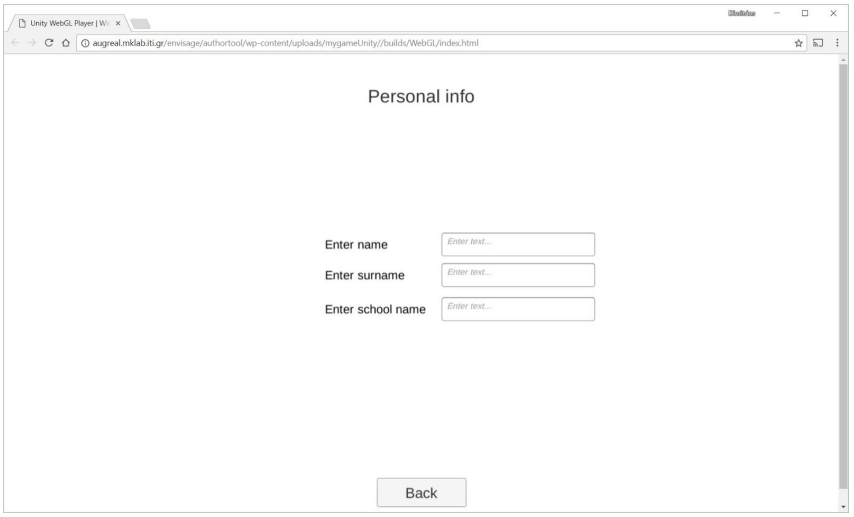


Figure S5.2: Login scene of the compiled game

On Main Menu press on “Settings” button and ensure that the following interface will be presented. Change the values to a desired configuration. The effects will take place on the educational scenes in the following screens. The “Windowed” option does not affect the web version of the game. Press “back” button.

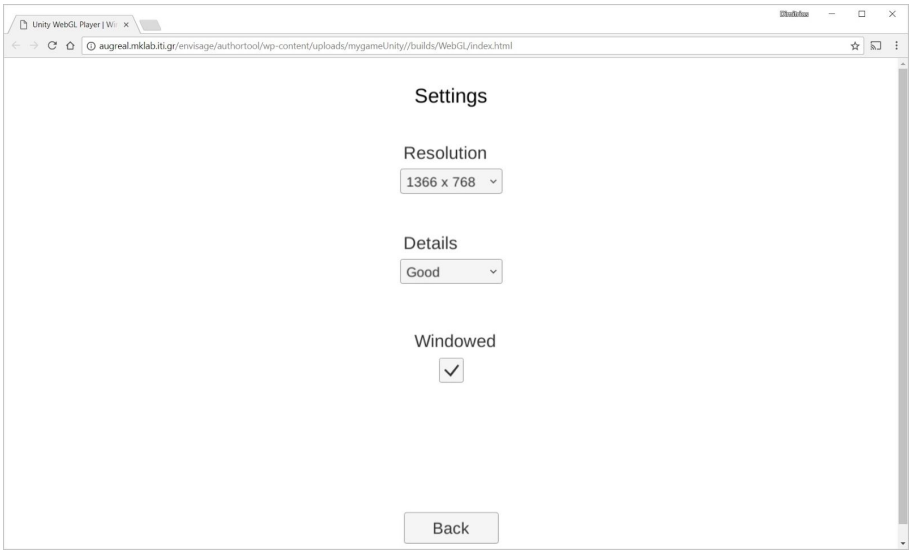


Figure S5.3: Options scene of the compiled game

Press on the “Credits” button and ensure that the following image is presented with the image and the text entered in Scenario 1. Press back to Main Menu.

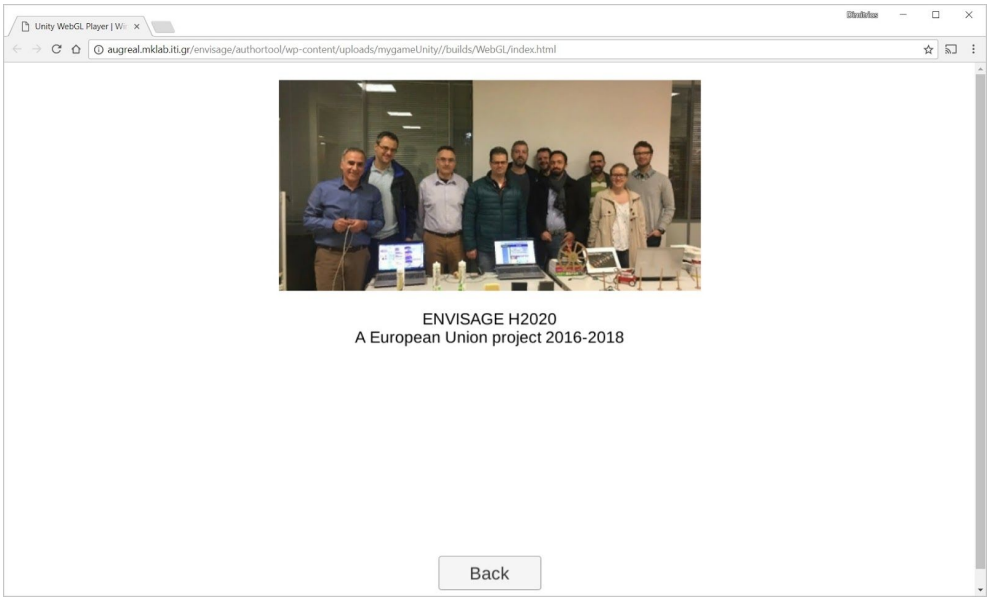


Figure S5.4: Credits scene of the compiled game

Press on “Play”. Ensure that the created scenes are shown correctly, namely the title, the description and the screenshot of the scene should be shown as in Figure S5.5. The title and description color should become black when hovering with mouse, and their background should turn white. When pressing with left mouse button on one scene, the scene should be loaded as follows.

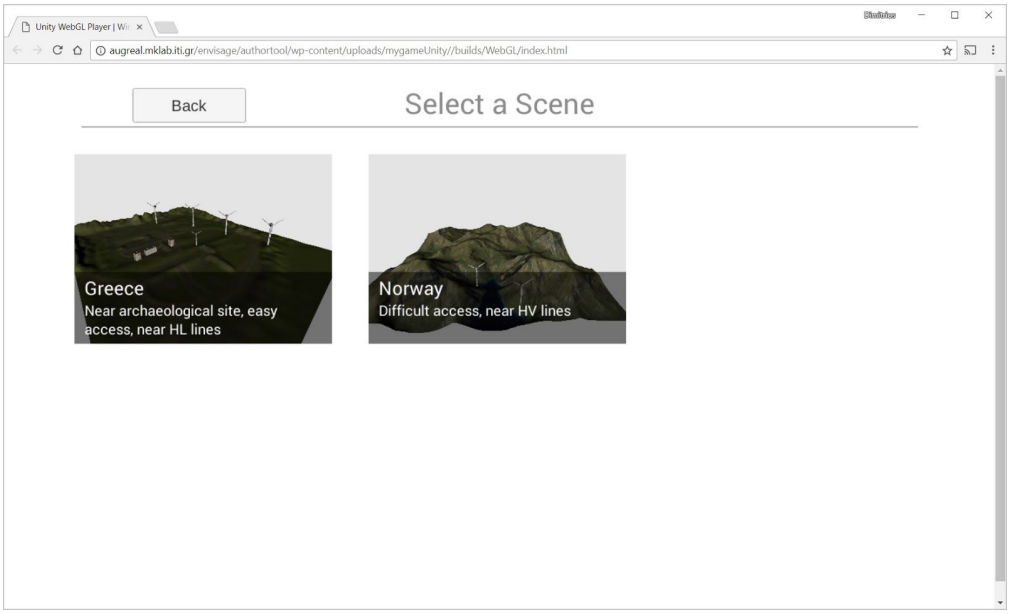


Figure S5.5: Login scene of the compiled game

The educational scene selected should be loaded. An example is shown in Figure S5.6. Ensure that the 2D gui elements and the 3D elements are shown correctly. The 2D elements are the bottom-left information panel of the game scores and energy levels, the clock, the the pause/play button, and the speed selection widget on the top of the page. The “Exit game” button terminates the current scene and goes to the reward scene. The 3D elements are the camera, the terrain, the building and the indicators for the turbines. First ensure that the 3D elements are placed on the correct position, with correct rotation and scale. Next ensure that the wind of the terrain is shown correctly in the information panel on the

bottom left. The values should change according to the mean and variance indicated on the right side of the panel (9±4). Next, the buildings (energy consumers) should be red indicating that they are underpowered. The billboards above them should present the mean and the variance of the energy consumed as entered in Scenario 1. Next, on hovering on the white circles, billboards for the candidate wind energy turbine should present the efficiency and the class of the turbine.

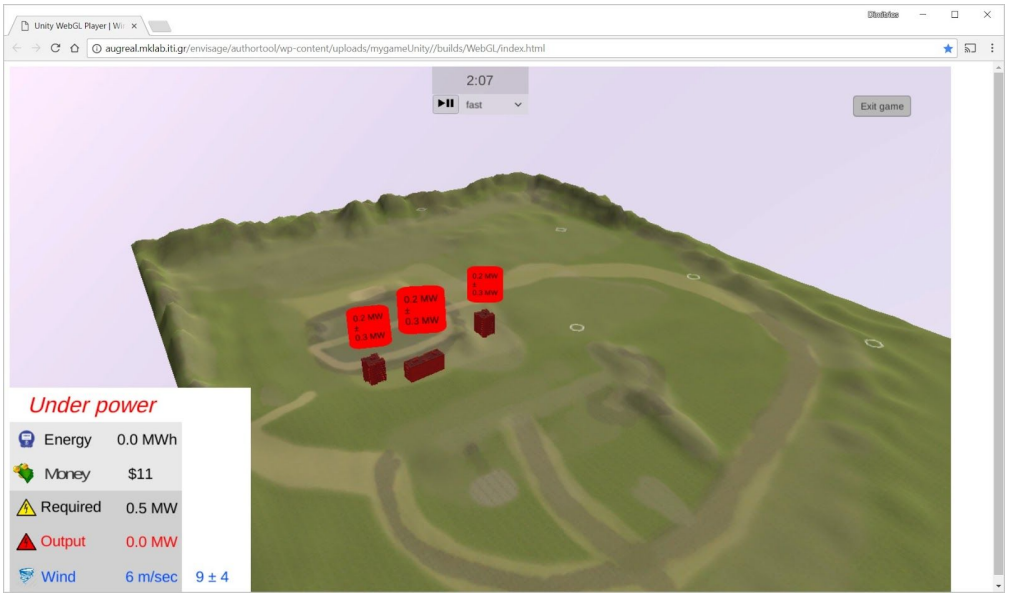


Figure S5.6: First scene of the compiled game

In a latter stage of the game, when the white circles were pressed, the turbines should produce energy. The indicators for this task is that buildings should change the color to transparent (correct power) or blue (over power), the money should increase, and the turbines should rotate. When hovering on the propeller of a turbine there should be an indication of “turn off” the turbine and the billboard of the turbine should present its current output.

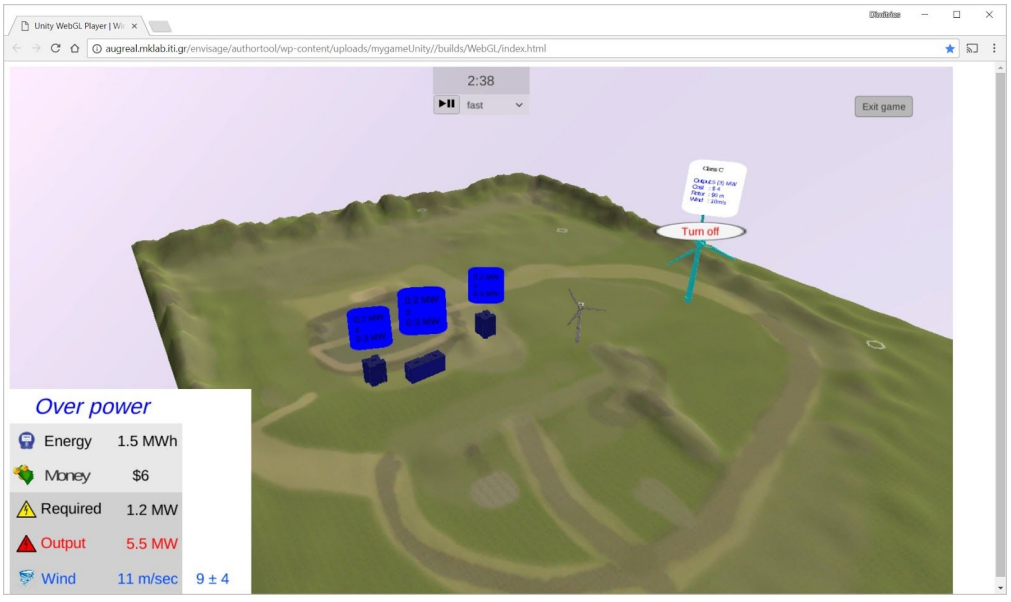


Figure S5.7: First scene of the compiled game at a different game state.

When pressing on “Exit Game”, the following scene should be presented. The scene should present reliably the minutes and seconds out of 24 minutes, where the state of the energy

balance was under-, correct-, and over -powered. A sentence should say which state was dominating during the scene played. By pressing Main Menu, the corresponding scene should be loaded.

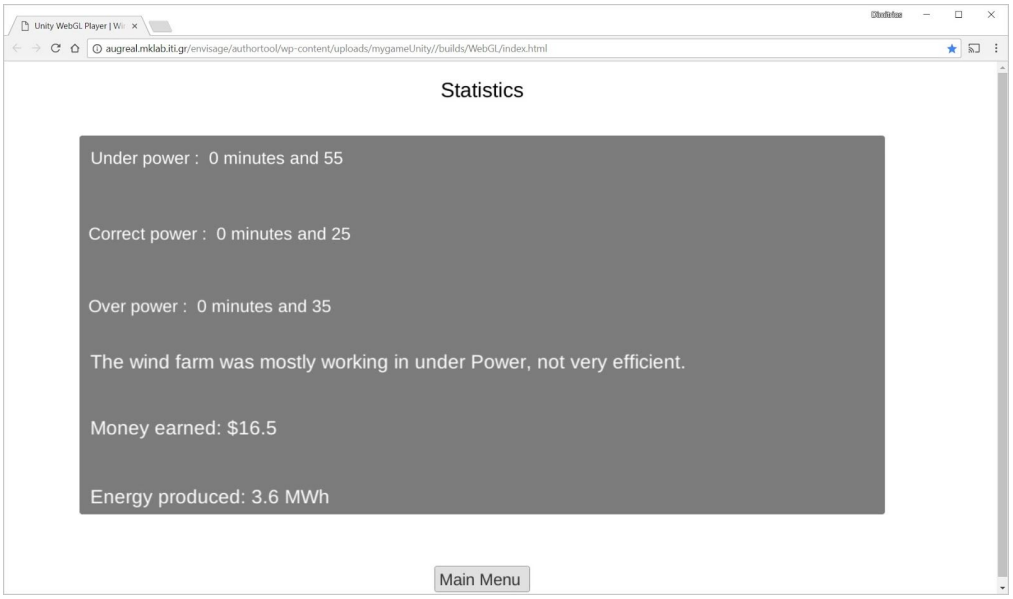


Figure S5.8: Reward scene of the compiled game

Scenario 6: Game analytics

Game analytics are loaded in the tab analytics next to the 3D editor of the scene as shown in Figure S6.1. The current version of the analytics iframe is showing analytics statistics of the previous versions of the games that were disseminated to the students. Currently only Wind Energy games are supported for inspection of analytics.

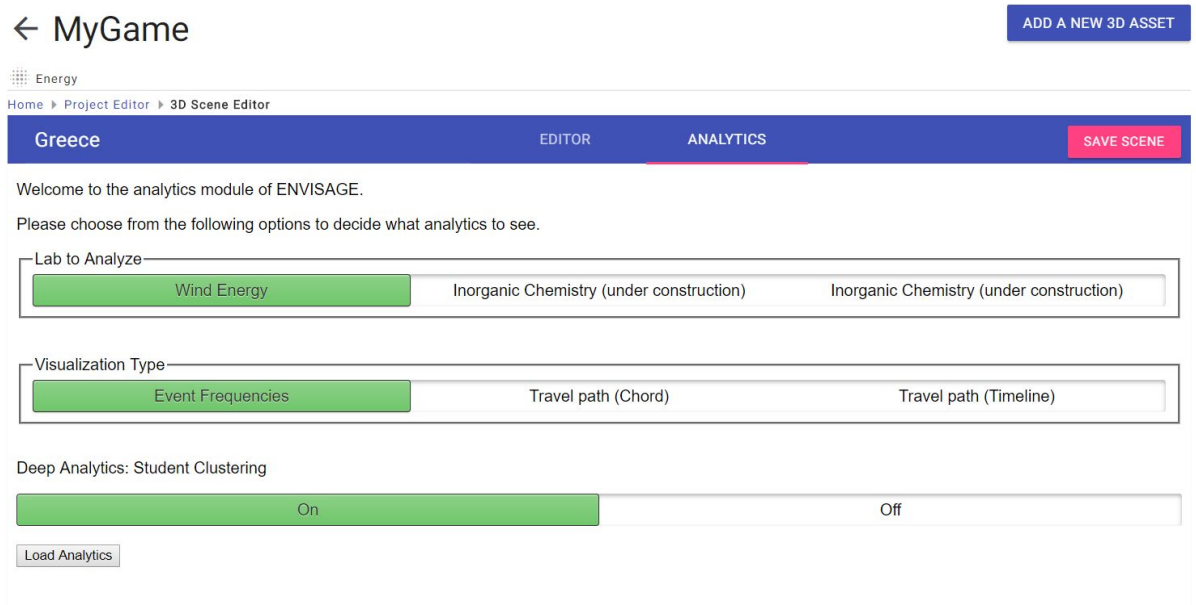


Figure S6.1: Game analytics central page contains options for displaying analytics.

Upon selecting the options for analytics and pressing “Load Analytics”, the interface of Figure S6.2 should be presented. The chord-chart presents all the actions during the game and the paths to the next action. As such, it represents all the travel paths in the selected sample, giving the educator crucial information about usage patterns in the game in question. The educator can filter the visualizations based on country the game was played and based on the time the events were received. The results can also be de-complexed when the user hovers on an action on the chord-chart as shown in Figure S6.3.

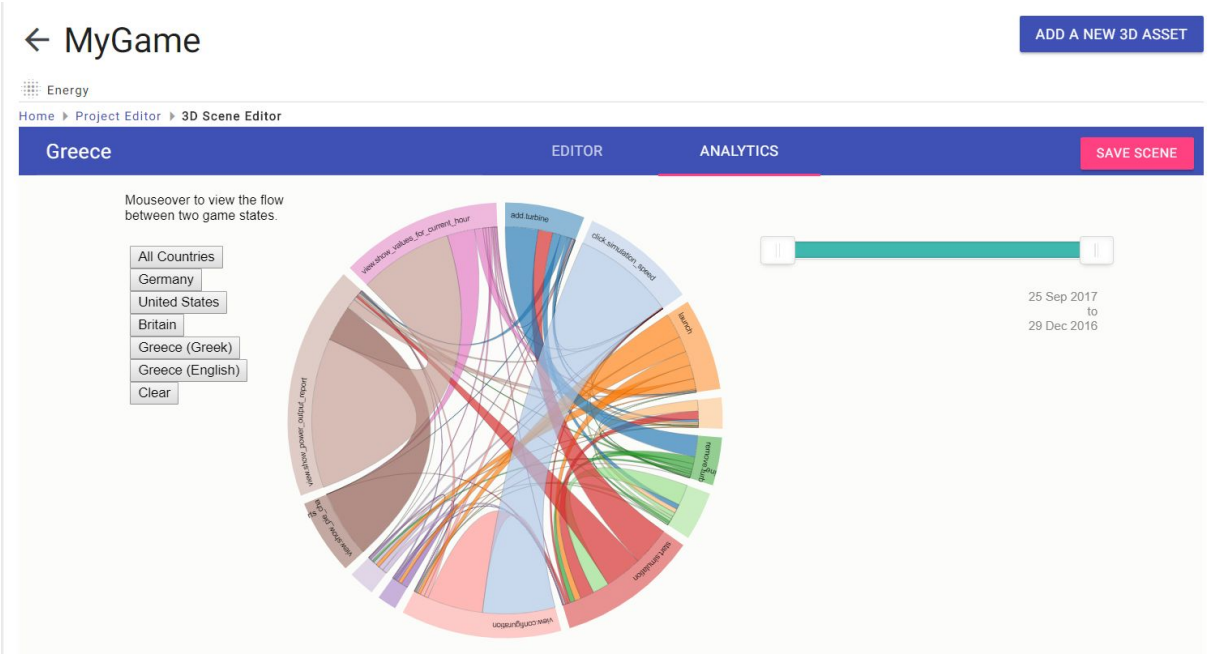


Figure 6.2: Pie-chart that presents games analytics as actions performed during the game.

Figure 6.3: An action is focused when hovering the mouse above it.

Summary

In this document we have presented the latest integration developments of the virtual labs authoring tool along with several scenarios for testing. In the following months, we will focus on increasing the quality of the generated games by developing more elaborated ones, by increasing the parameterization and the capabilities of the analytics tab, and by developing a new templated for the chemistry case.