Co-funded by
the European Union

ICT solution:

„Demo version of a training application that utilizes virtual reality (VR) technology"

1. System description and purpose of the tool

A training application using virtual reality (VR) technology is an advanced educational tool that combines interactivity, immersiveness and realism to ensure the most effective learning process possible. The system is designed to provide users with an intense experience that enables them to learn and improve their skills in a virtual environment. Thanks to the use of VR technology, the application allows full immersion in the training process (the so-called immersion effect), eliminating ambient distractions and guaranteeing a high level of involvement of the participants in the process in the taught content.

The application utilizes a multiplayer gameplay mode, providing the possibility for multiple users to share training scenarios at the same time, using real-time gameplay state synchronization. This innovative approach to the way of realizing training solutions with the use of VR tools, gives the possibility of realizing the process simultaneously for groups of training participants, and not only for an individual user, which provides unprecedented opportunities for the realization of training processes. The solution provides the opportunity for participants to join a training session from anywhere in the world (provided they have access to a stable Internet connection), which is of great value in terms of system interoperability and, with the right approach, can open up completely new opportunities for educational and professional mobility.

According to the project assumptions: the application is a demonstration version of solutions of this type, presenting a good example of how to implement the training process using VR/AR tools. Its primary goal is to promote the use of VR and AR technologies in the training process and demonstrate

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

the attractiveness and effectiveness of such an approach. With this in mind, the solution therefore fulfills training functions in two ways:

- Direct - giving the opportunity to implement training scenarios included in the application and acquire competencies through them, as well as acquire microcredentials, which constitute a proof of acquired competencies.
- Indirect - by educating and informing stakeholders from the following groups: training and education providers, institutional decision makers in the field of education, labor market representatives; on possible ways to implement ed-tech solutions, good practices, and the potential behind the use of such an approach.

The application will also be used during pilot implementations carried out during the project, such as workshops for end-beneficiaries, planned for implementation under Task: T.2.4.

1.2 Overview of the technologies used to create the tool

1.2.1 Engine

The VR training application was developed based on the **Unity** engine, which provides a comprehensive environment for developing XR-enabled interactive applications. Unity offers a wide set of tools to implement advanced user interaction mechanisms in a VR environment, using the **C#** programming language and component system to manage objects and scenes. The built-in **PhysX** physics engine provides realistic simulations of object interactions, while the dedicated **Input System** and **XR Interaction Toolkit** enable the implementation of controls, motion tracking and manipulation mechanisms in 3D space.

1.2.2 VR standard

In order to support the VR standard, the application uses **OpenXR**, which acts as a universal interface that allows the application to integrate with a wide range of VR hardware platforms. OpenXR provides a unified API to support devices such as Oculus, HTC Vive and Windows Mixed Reality, eliminating the need for dedicated SDKs (Software Development Kits) for individual manufacturers. With OpenXR, an application can run on different hardware ecosystems without modifying the source code, affecting its scalability and ease of deployment on different VR platforms. With this in mind, however, it should

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

be noted that despite the fact that the OpenXR standard supports multiple VR platforms, it is still necessary to optimize (port) applications individually for each manufacturer's devices. This is due, for example, to the difference in the way the functions of manipulators (controllers, which have different shapes and different mapping of buttons and other functional keys, such as analog knobs, triggers, etc.) are implemented. While working on the application, it was decided to optimize its functionality for Oculus-branded devices (currently Meta). This decision was conscious and was dictated by the fact that the devices of this manufacturer are the most widely available, are used by the largest number of users and are the cheapest of the VR solutions currently available on the market. As a result, the solution is able to reach the highest number of potential users, affecting the reach of the application and the ease of access to it.

1.2.3 3D models

Blender software was used to create 3D models, which offers advanced tools for modeling, sculpting, texturing and animating 3D objects. Blender allows models to be exported in Unity-compatible formats such as FBX and glTF, allowing for seamless integration of graphics assets into the engine. The use of advanced geometry optimization techniques, such as vertex reduction, allows models to maintain high quality while minimizing the impact on rendering performance in VR environments.

1.2.4 Graphic materials

Adobe Photoshop software and **Adobe Creative Cloud** suite were used in the process of creating graphic materials and textures, which enabled precise PBR (Physically Based Rendering) texture editing, elevation mapping, and light emission.

1.2.5 Interoperability

During the development of the solution, special emphasis was placed on using open source, free and readily available solutions. The assumption that guided the development team was based on the need to preserve the interoperability of the solution and to demonstrate the availability of technological solutions for the development of comparable solutions and their replication.

Of the technological solutions used in the development process (described above and (more extensively) later in the document (in the section on system architecture), only a few require a paid subscription (i.e., applications included in the Adobe suite, and the Vivox solution - utilized for the facilitation of voice communication via a server, for the implementation of multiplayer gameplay; with Vivox being a paid option only when the number of users exceeds 20,000).

References to the technical documentation used during the implementation of the system:

- Unity - https://docs.unity3d.com
- Blender - https://docs.blender.org
- OpenXR - https://www.khronos.org/openxr/
- Adobe Photoshop - https://helpx.adobe.com/photoshop/user-guide.html
- Programming language C# - https://learn.microsoft.com/en-us/dotnet/csharp/
- Oculus Developers - https://developer.oculus.com

1.3 System usage

1.3.1 Technical requirements

The VR system is designed to be used from a VR platform. In order to use the functionality of the system, it is necessary to have a compatible HMD (Head Mounted Display) device, compatible with the requirements of the system. As previously mentioned, the system has been optimized for Meta's solutions, so it will work most efficiently and comfortably on the brand's devices (all currently supported models, such as Meta Quest 2, Meta Quest 3, Meta Quest 3S). The solution SHOULD also work properly on the brand's discontinued lines (Oculus Rift S, Oculus Quest, Meta Quest Pro), however, the lack of current manufacturer support may make it partially incompatible with some functionalities of the technological solutions that were used during the design of the system.

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

The solution SHOULD work properly on any device supported by the OpenXR standard, but due to the reason described in section 1.2.2 above (the need to optimize the system for use on each manufacturer's solutions), use may involve discomfort, and some functionality may not work properly.

Minimum hardware requirements of the HMD device to work with the solution:

- Audio: built-in speakers, microphone,
- Connectivity: Wi-Fi
- Tracking: 6 DoF
- Built-in memory: at least 128 GB
- RAM: 6 GB
- Two touch controllers
- Compatibility with OpenXR standard

In order to ensure comfortable conditions for the use of the solution, it is recommended to use an HMD kit that meets the following parameters (additional requirements - recommended):

- Screen resolution: 1832x1920 pixels per eye
- Refresh rate: at least 90 Hz
- Horizontal field of view (FOV): at least 110°

Additional recommendations:

To ensure the best user experience, it is recommended to use a headset connected to a stable Internet connection with high bandwidth (especially when implementing multiplayer gameplay).

The technical requirements described above are objectively low, and the use of the solution does not require high-end hardware solutions. Bearing in mind the need to build a solution with easy accessibility, efforts have been made to optimize the system in terms of consumption of technical resources of output devices. The solution itself, meanwhile, is scalable and will look better on higher-end devices.

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

The application has been optimized for autonomous operation on standalone headsets, which eliminates the need for additional computer hardware.

1.3.2 Access to the application

According to project assumptions, the application will be accessible from the register of training services for the Modern Business Services (MBS) sector, which is a separate product of the project, and which is available at: https://mbssapp.vccsystem.eu/en.

Due to the way in which the VR application system and the registry system are integrated with each other (e.g., in terms of handling functionality related to the granting and collection of microcredentials), the user account in both systems must be linked to each other. Therefore, the application will be available only to registered users of the registry (with a user account of individual user profile). Use of the application by an unregistered user is not possible, due to the need to identify the user for the issuance of microcredentials in the form of digital badges, earned from within the application (the identification of the person to whom the credential is issued is a necessary, integral part of the data encrypted in the credential, enabling its validation).

- A user logged into the registry system can find (from the user profile view) a tab relating to access to VR application. In this space will be located:
- Description of the application,
- Technical requirements for the devices used to operate the application,
- Instructions on how to install the application on the device,
- An instructional video on how to use the system,
- Access data for the application.

As mentioned above - the user must be logged into the VR application in order to use it. Access data to the VR application is granted within 24 hours of registering an account in the registry system. After launching the application, the user is confronted with a login window where access data, accessible from the registry, must be entered.

1.3.3 Operating the system

A person using a VR headset has a unique way of controlling and interacting with the world. In VR mode, locomotion is achieved using head and limb movement, as well as horizontal movement (walking, leaning). The method of locomotion used in the application utilizes 6 DoF (six degrees of freedom), instead of the 3 DoF common in low-quality solutions. This allows for much more user interaction with the virtual environment and exploration of the presented world while retaining user immersion.

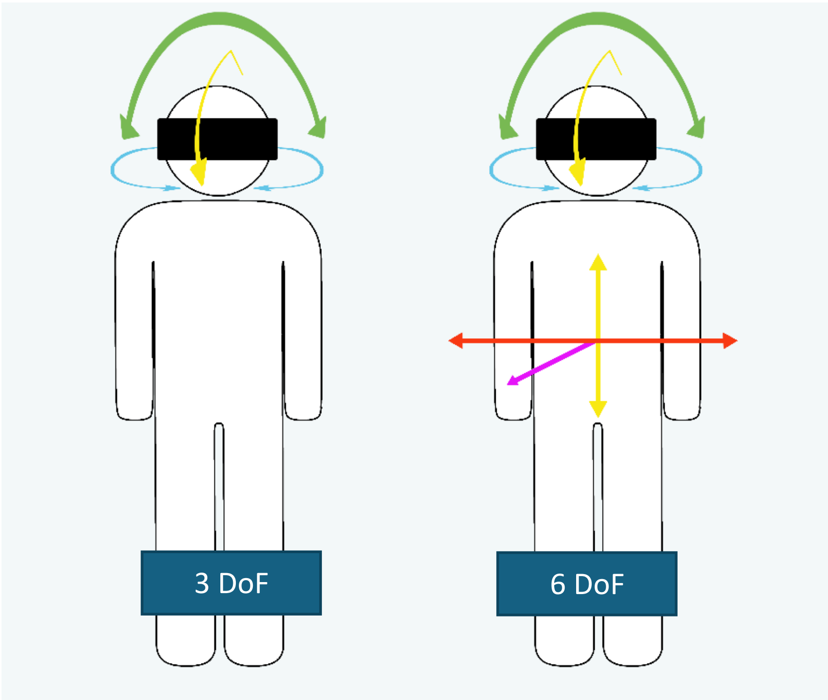

Figure 1. Scope of motion tracking in 3 DoF and 6 DoF systems

The user's movement in the real world is transferred 1:1 to the simulation space. In addition to this (for reasons of user safety and limited, in principle, gameplay space), the application implements locomotion using the functionality of manipulators (movement using analog sticks). In order to reduce the occurrence of symptoms of the so-called simulator sickness, for the purposes of indirect locomotion (using manipulators), an option was used to parameterize the movement function, giving the possibility to rotate in a smooth or incremental manner and to move in a smooth manner or by means of teleportation functionality (the user disappears and appears automatically in the place indicated by the manipulator pointer).

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

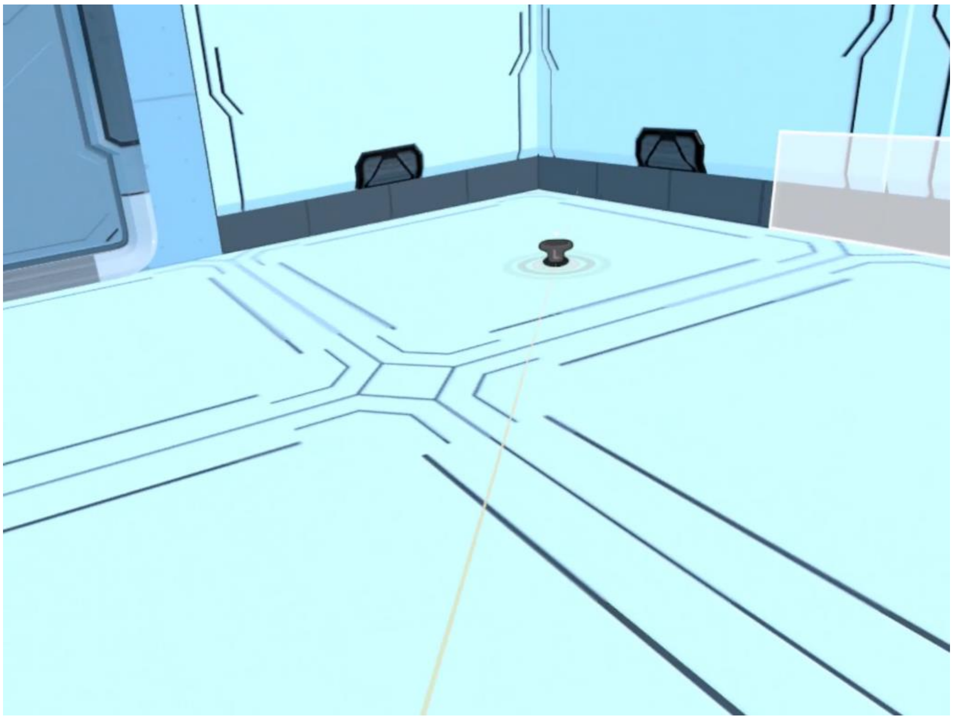

Figure 2. An example of locomotion with teleportation mechanism - visualization of the analog stick indicates the teleportation destination

Tracking of the user is carried out using sensors that are part of the VR kit's hardware and mounted in the headset and manipulators. The application has been parameterized to utilize 3 tracking points (a solution universally utilized by VR headsets).

The user's silhouette is reflected in the application in the form of a virtual persona (avatar). This allows users in multiplayer mode to see each other and interact with each other.

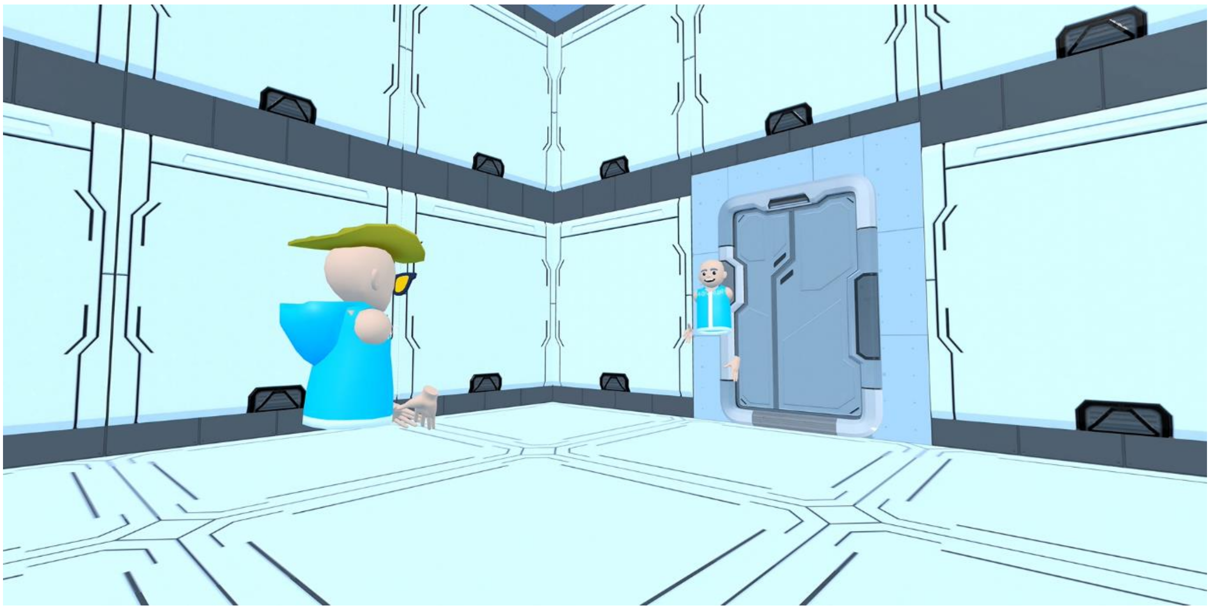

Figure 3. Avatars of two users in the gameplay environment

Interacting with objects within a VR simulation requires physical movement. Grabbing objects, lifting, moving, dropping and throwing them requires reaching for the object with the hand (there is no ability to grab and move objects at a distance) and grabbing it using the manipulator function (hand tracking is not supported).

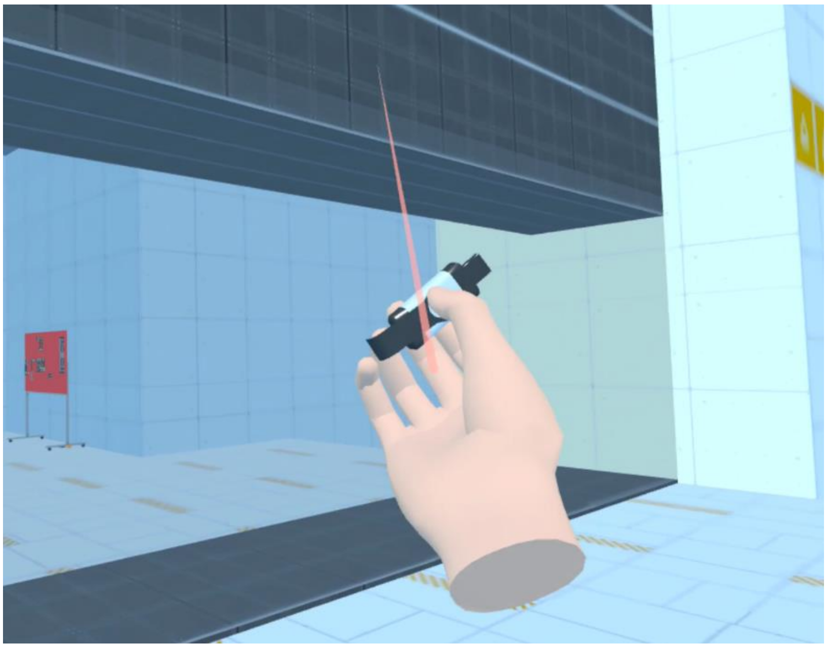

Figure 4. An example of grabbing a virtual object

Interactions with static objects (pushing buttons, moving sliders, turning a crank, etc.) are implemented in a similar way (using physical movement).

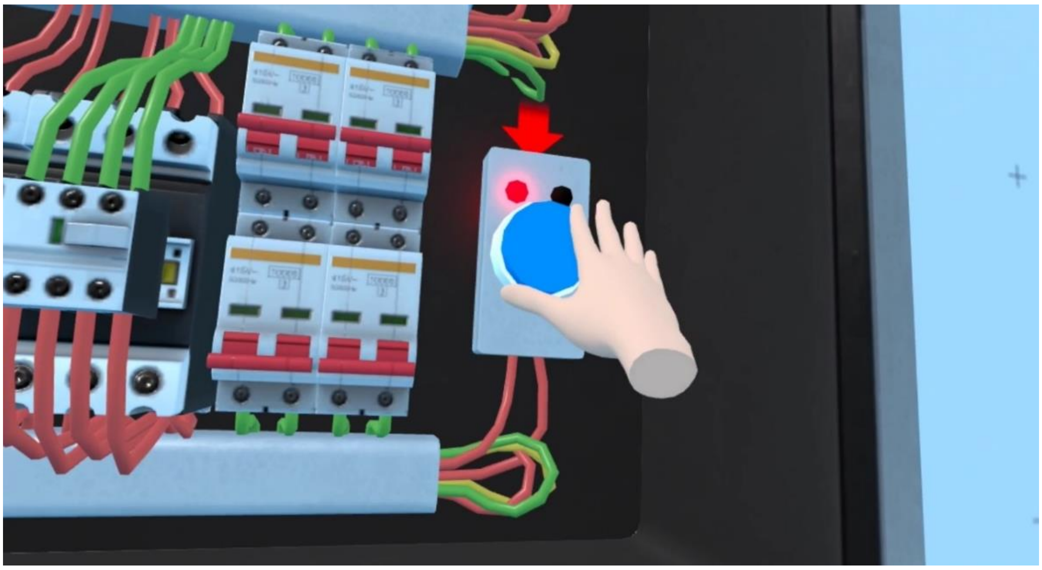

Figure 5. An example of an interaction with a static object - pressing a button

The application provides for the possibility of using objects on other objects to produce the desired effect (e.g., placing objects in holes, hammering a nail with a hammer, inserting a pen-drive-type data stick into a USB port, etc.).

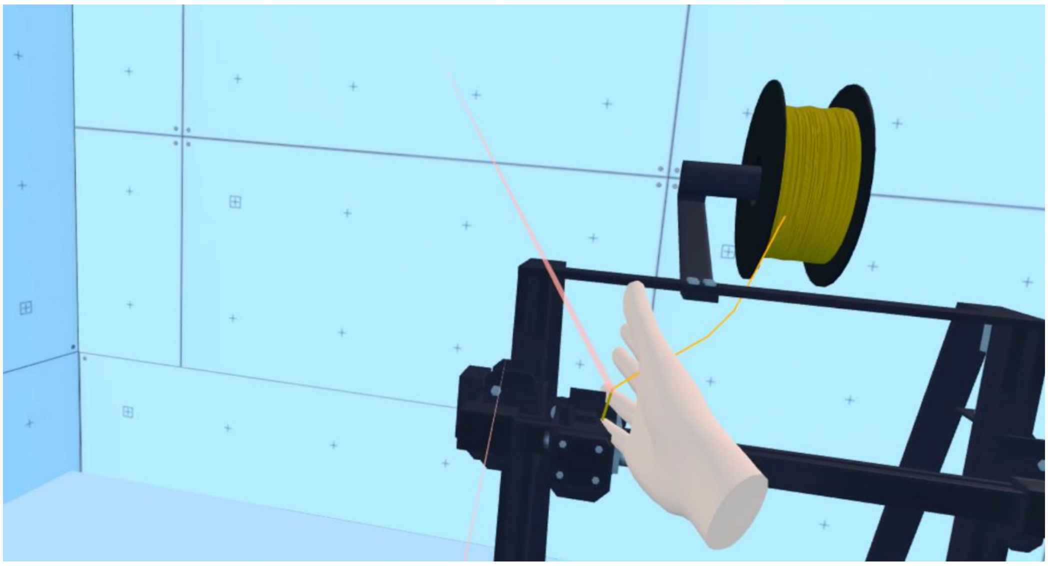

Figure 6. An example of the interaction of two objects in use - the insertion of the filament into the extruder of a 3D printer

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

Co-funded by
the European Union

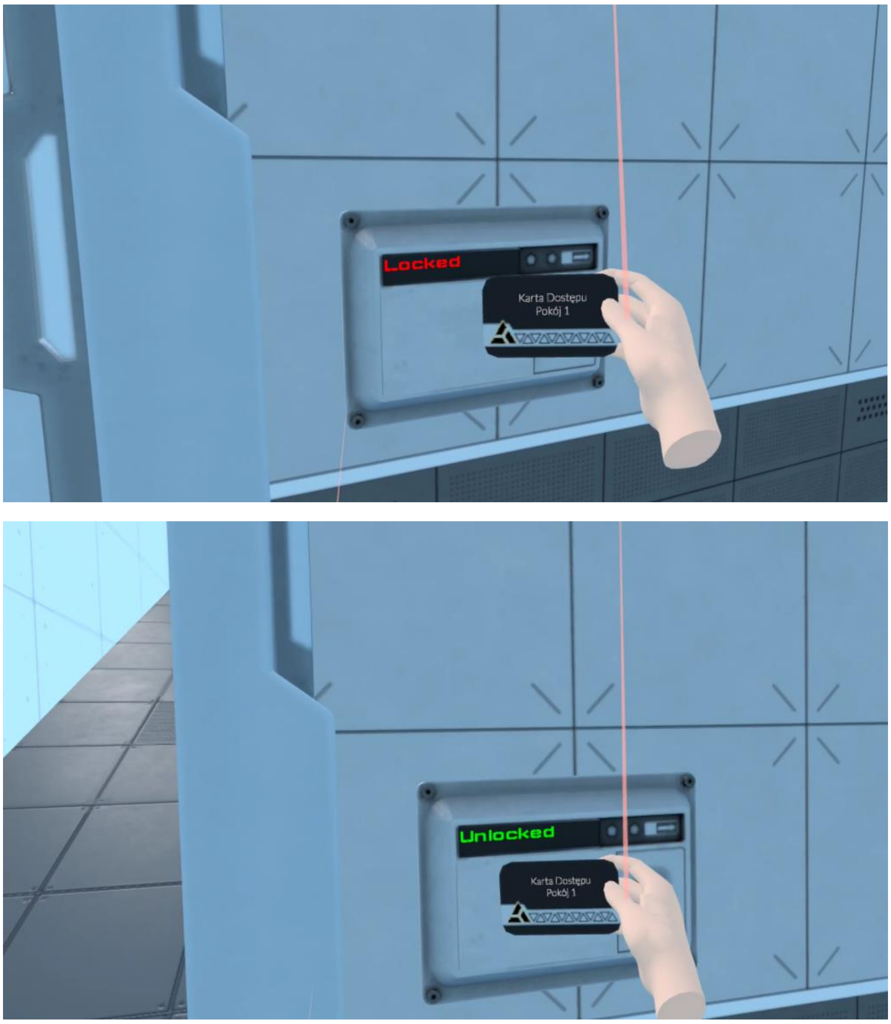

Figure 7. An example of the interaction of two objects in use, causing an effect in the gameplay space - holding a card to a magnetic lock causes the door to open

The application implements simulation of the physics of space and objects within the basic principles of the standard model (such as gravity, i.e. dropped objects fall to the floor, etc.). All interactions implemented within the simulation space are designed to provide the highest possible level of user immersion.

UI elements are operated using virtual laser pointers, the position of which is linked to the position of the manipulator (the pointer is used by pointing at objects with the hand). This is a standard solution for UI handling, used, for example, in the operating systems of VR devices. This solution was used

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

because of its versatility, recognizability and ease of use. The indicators can be activated and deactivated using the manipulator button.

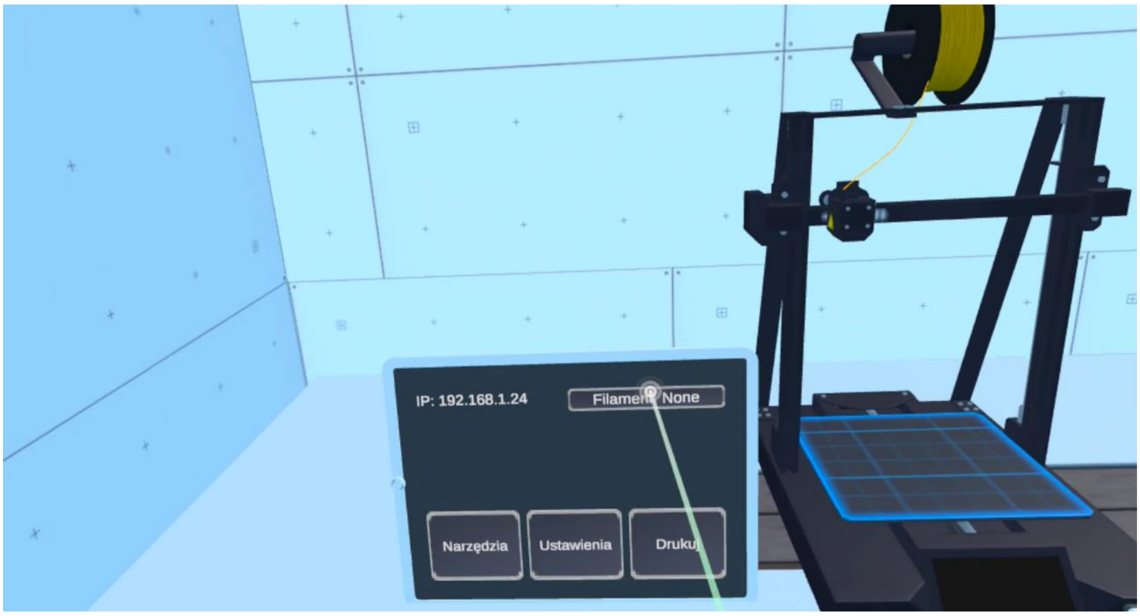

Figure 8. An example of interaction using a pointer

The usable gameplay space is predefined and limited. The user cannot go beyond the boundaries of the playable world, which is ensured by coliders placed within easily recognizable gameplay boundaries (e.g., on room walls) that inhibit movement. This is a desirable effect - typical of interactive 3D spaces.

1.3.4 Separation of Interaction and UI interaction

In order to provide more flexibility and better management of user permissions, the interaction system in the VR application has been divided into two separate interaction modalities: ***Interaction*** and ***UI Interaction***. With this solution, it is possible to specify precisely what actions a user can perform in the training environment, which is especially important in multiplayer mode.

Interaction includes all activities involving physical manipulation of objects in the VR world. The user can pick up objects, move them, place them in specific zones, and perform other interactions that require active action in space (as described above). If this feature is restricted, the user can still move within the gameplay area, but cannot affect interactive elements of the environment.

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

UI interaction refers to the use of user interface elements, such as buttons (not to be confused with buttons that are objects), displaying additional information, changing settings or adjusting options. Even if the user is not able to physically interact with the objects, they can still operate the interface elements, allowing them to passively participate in the training scenario.

Separating the two modes of interaction is important for the application's functionality, especially in a multi-user environment. It allows flexible adjustment of user permissions according to their role in the scenario. For example, in multiplayer mode you can specify which users will actively participate in the training by manipulating objects in the simulation space, and which will remain passive observers, having access only to the user interface. Such a solution facilitates the organization of training sessions, allowing the creation of roles tailored to different levels of participation.

Thanks to this division, the application provides better control over interactions, allows for precise management of permissions, and allows for more flexible customization of the user experience according to the user's needs and functions in a given scenario.

1.3.5 Experimental functions - alternative interface platforms

At the moment, the development team is working on the creation of an alternative version of inetrface, which will provide the possibility to use the application from the desktop platform. This largely defeats the purpose of using VR technology, since an integral part of the technology is the hardware associated with its use, namely the output device in the form of a VR headset.

Nevertheless, the introduction of such a version of the interface will create additional possibilities for the way the training process is implemented. Using the desktop interface, the trainer can act as an observer of the trainees, who implement the activities of the training scenario from the VR platform. Such a solution has its application in assessments and validation of performed activities. In addition, this can be a good solution for training groups that do not have a sufficient number of VR sets or adequate space to implement gameplay in the VR environment for the entire group at once. In such a situation, part of the group can implement activities in the virtual environment, while the rest can act as an observer of the performed activities (e.g., for the purpose of instructional implementation).

With this alternative version of the interface, locomotion will be implemented using a keyboard with a mouse or touchpad. Appropriately mapped keys will allow movement in the forward-backward and left-right axes, as well as to change the perspective elevation. The control will also allow camera

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

rotation (the de facto perspective of the user, since the observation point with this type of interface will also be first-person).

Unlike using the system in VR mode, the desktop interface does not have restrictions on movement within the simulation (because the coliders are configured to restrict the movement of the tracking point placed in the headset by objects and obstacles, and in the case of the desktop interface we do not use the headset). As a result, the user can penetrate walls and other obstacles. This is a side effect of using software dedicated to VR technology on a device that is not directly compatible with it.

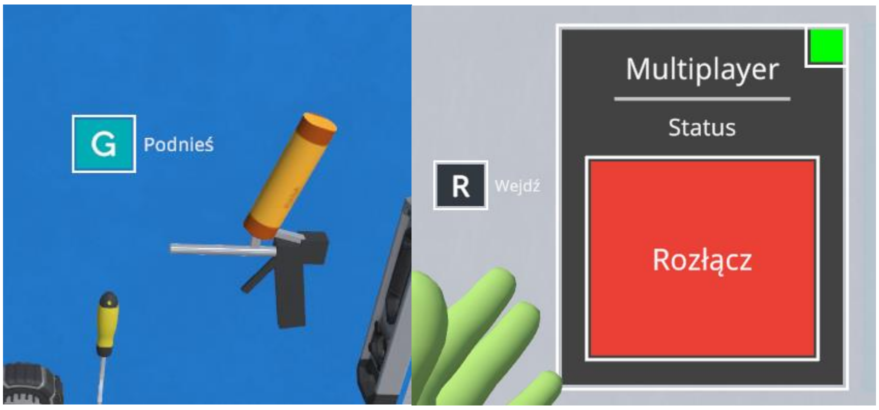

Figure 9. Example of prompts referring to the use of computer keyboard keys - experimental application desktop interface

Alternative options for running the application interface on mobile devices are also being explored, which essentially works in a comparable way to the desktop interface. The difference lies in the way the controls are implemented, which in the case of mobile devices will be based on the use of virtual keys, displayed on a touch screen.

The described experimental functions were not intended to be implemented as part of the project. They were designed to optimize the solution and maximize its educational potential. This type of experimental approach is justified by the fact that one of the primary functions of VR application is to educate and inform stakeholders about the potential capabilities of the technology.

Alternative versions of the interface do not represent working functionality at the time of publication of this document and will be introduced with subsequent development versions of the software.

1.3.6 Mapping of controller functions

The figure below shows the mapping of the manipulator buttons used in the use of the application. The prompts presented below, which describe how each button works (a kind of tutorial), are called up and hidden using the handheld UI function (a menu called up by one of the manipulator buttons). By default, the manipulators are visualized in the VR environment in the form of hand simulations (as can be seen in several figures in earlier sections of the document). Initiating a button mapping instruction changes the way the controllers are visualized from a hand simulation to a controller shape simulation view (as seen in the accompanying figure); hiding the instruction changes the way the controllers are visualized again - restoring the hand simulation visualization.

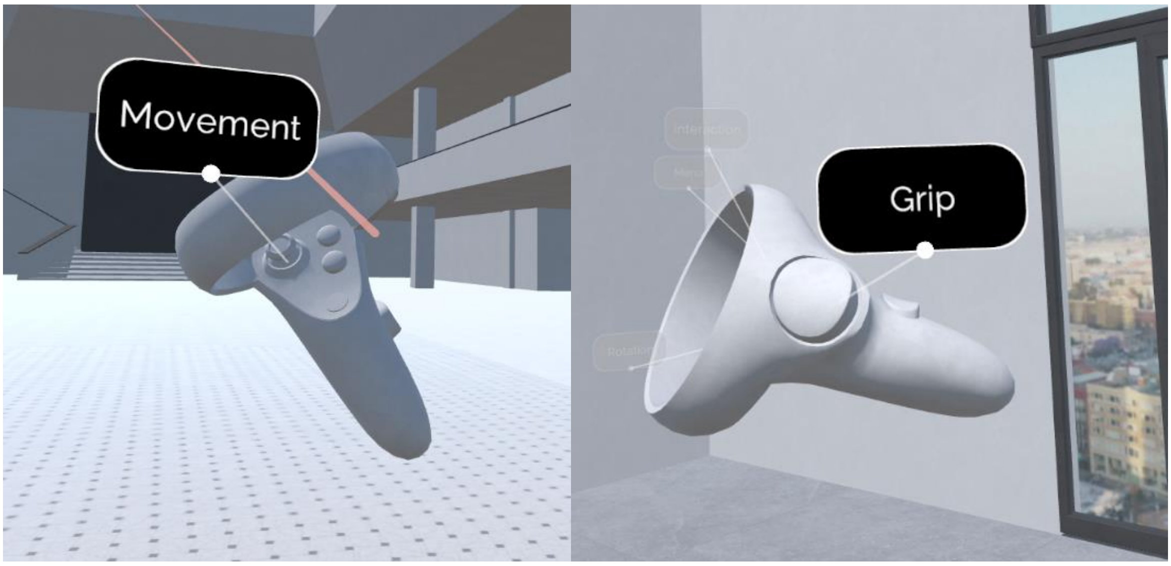

Figure 10. Visualization of instructions on button mapping - using a button displays its description

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

2. System architecture

2.1 Introduction to the system architecture

The architecture of the VR application is built on components that guarantee modularity, flexibility and scalability. The project is based on a three-layer architecture model:

- The presentation layer (frontend).
- The business logic layer (backend).
- The data layer (connection to the server).

Each of these components works together through an API, making it easy to integrate new features and external systems.

2.2 The presentation layer - Frontend

The frontend of the system is created by using the built-in UGUI solution in the Unity environment. **UGUI** (Unity UI) is a user interface system integrated with Unity that enables the creation of dynamic and interactive UI elements in applications. It is based on Unity's component system and uses an object hierarchy to organize the interface structure. User interactions are handled by **EventSystem**, which manages user actions, such as mouse clicks, screen touches and control with the VR headset controller. EventSystem works with event handling components such as Button (**OnClick**) and **EventTrigger** to respond to various user actions. To support UGUI elements with VR controllers, overlays were created for the project, allowing free interaction with objects.

2.2.1 UI components

2.2.1.1 Text cards

Interface elements located in the VR environment. Interaction with the elements follows the rules typical of ***UI Interaction*** (see Section 1.3.4 Separation of Interaction and UI Interaction). Each text card presents an informational context in the form of text.

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

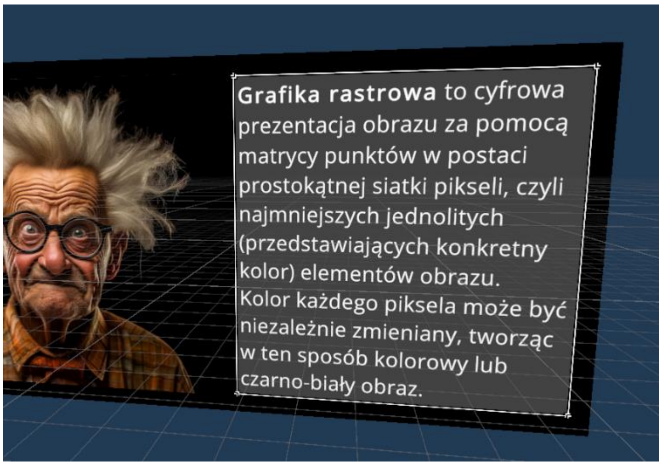

Figure 11. An example of a text card element - a screenshot taken from the development environment (Unity)

2.2.1.2 Visual responsive text cards

Interface elements located in the VR environment. Interaction with elements follows the rules typical of ***UI Interaction*** (see section 1.3.4 Separation of Interaction and UI Interaction). Visual responsive panels remain invisible until the user directs his or her gaze to them (effectively - approaches the element at a distance that triggers its display, while facing it). Alternatively, it is possible to call them up by pointing a pointer toward them. Elements contain information context in the form of text. Cards are visible for a declared time and then disappear.

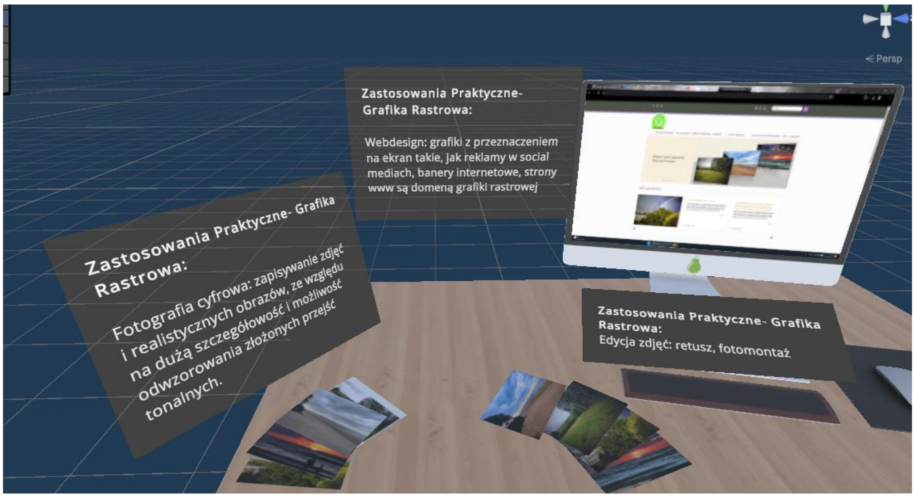

Figure 12. An example of a text card element that reacts to gaze - a screenshot taken from the development environment (Unity)

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

2.2.1.3 Graphical panels

Interface elements located in the VR environment. Interaction with elements follows the rules typical of ***UI Interaction*** (see Section 1.3.4 Separation of Interaction and UI Interaction). Elements of this type display an image or series of images, changing the rendered graphics as the application runs.

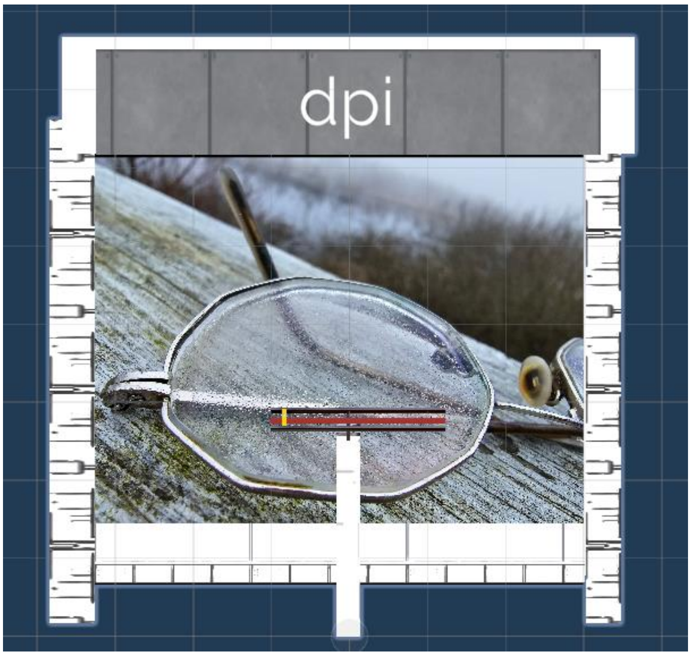

Figure 13. An example of a graphical panel element - a screenshot taken from the development environment (Unity)

2.2.1.4 Physical manipulators

Interface elements located in the VR environment. Interaction with the elements follows the rules typical of ***Interaction*** (see Section 1.3.4 Separation of Interaction and UI Interaction). By means of physical interaction with static objects in the environment (objects that can be used but cannot be moved, lifted or thrown), such as grasping, moving, rotating, the user can influence the VR environment and control the gameplay to a defined extent. Examples of objects of this type are sliders, rotating cranks, levers, etc.

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

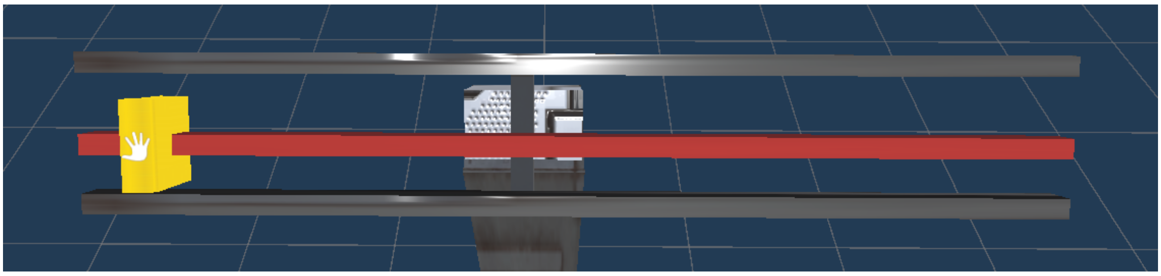

Figure 14. An example of a physical manipulator element, slider type - moving the slider to the right and to the left creates an effect in the VR environment

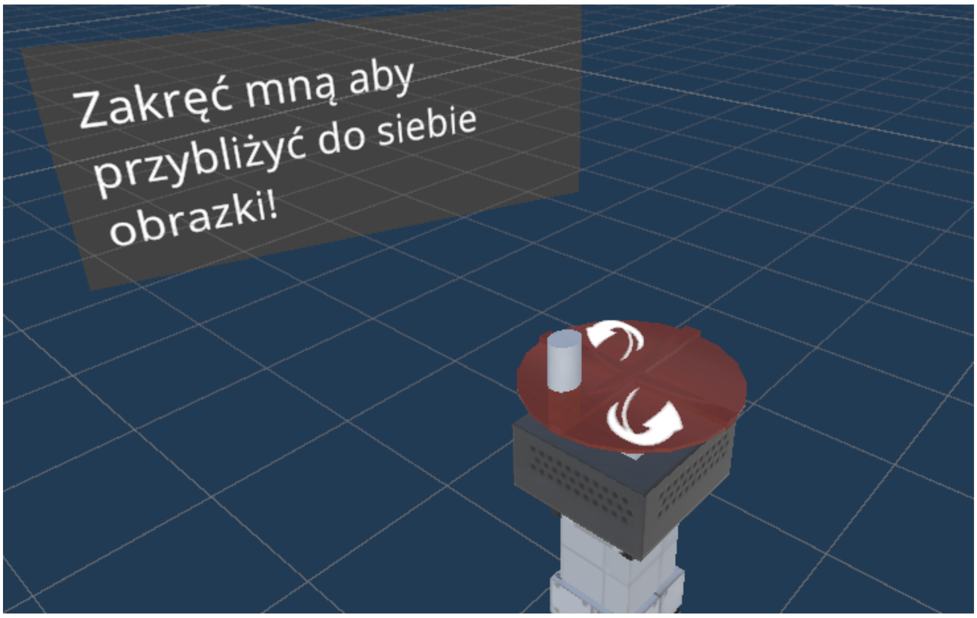

Figure 15. An example of a physical manipulator element, of the rotary crank type - turning produces an effect

2.2.1.5 Buttons & keys

Interface elements located in the VR environment. Interaction with the elements follows the rules typical of ***Interaction*** (see Section 1.3.4 Separation of Interaction and UI Interaction). Elements of this type are static objects located in the simulation environment that can be pressed to produce an effect.

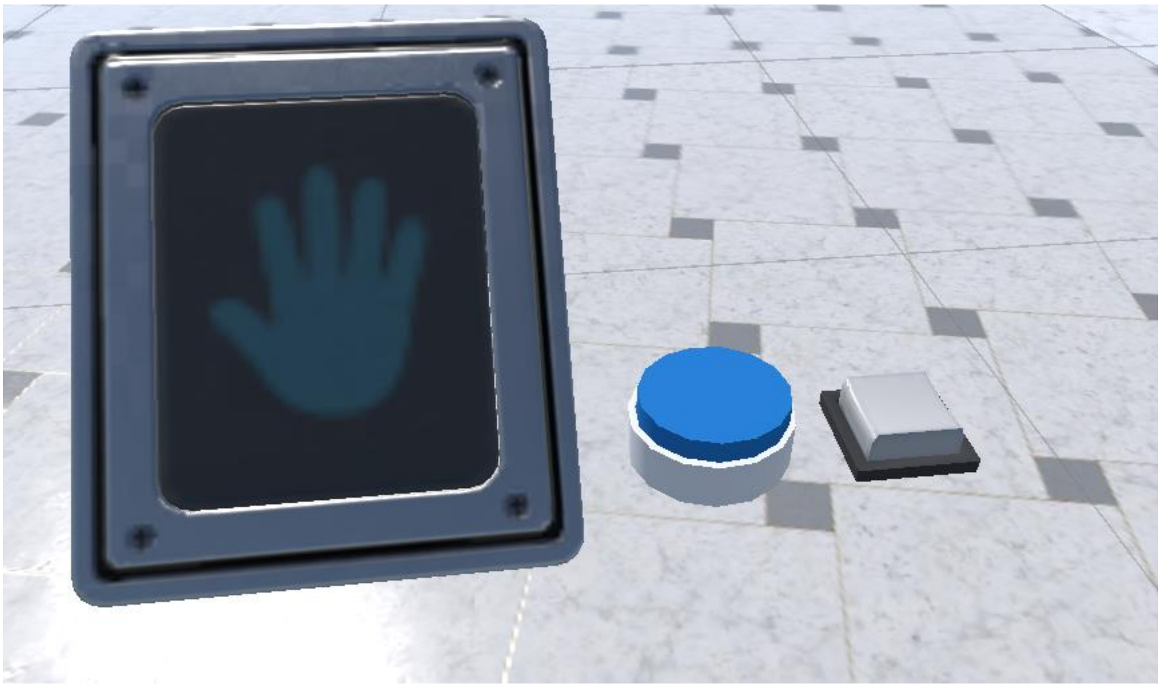

Figure 16. Examples of buttons and keys appearing in the application

2.3 The business logic layer

The business logic of the application has been implemented in C#, ensuring high performance and readability of the code, as well as ease of maintenance and expansion of the system. The structure of the application is based on a modular architecture, allowing flexible management of individual components and their independent development. Thanks to the modular approach, it is possible to separate the layers responsible for data processing, operational logic and interaction with the presentation layer, which increases scalability and facilitates the implementation of new functionalities.

The backend architecture is designed for extensibility and compliance with software engineering best practices. The use of the C# language allows integration with a wide ecosystem of technologies, including frameworks for building server-side applications and tools to facilitate data flow management and processing. The modular structure of the system provides a high level of encapsulation, allowing easier testing of individual components and minimizing potential errors resulting from dependencies between modules.

Using a layered approach, the backend effectively separates business logic from the data access and user interface layers, allowing for greater transparency and better source code management. The

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

modularity of the architecture also enables easier adaptation to changing business requirements and streamlines implementation processes. The system is designed with performance and scalability in mind, enabling efficient processing of large amounts of data and support for multiple simultaneous operations in a production environment.

2.3.1 Backend layer components

2.3.1.1 .NET Framework

The backend layer is based on **.NET Framework** technology, providing a stable and efficient runtime environment for applications. With support for multithreading and an extensive ecosystem of libraries, .NET enables efficient resource management and implementation of complex logical operations. The Framework supports **Just-In-Time** (JIT) compilation mechanisms, which optimizes code performance in real time, and also ensures compatibility with a wide range of integration technologies, enabling extensibility and adaptation to new functional requirements.

2.3.1.2 OpenXR

The backend has been designed with full support for the OpenXR standard, providing a unified and abstract layer of access to virtual reality and augmented reality technologies. OpenXR eliminates the need for dedicated SDKs for individual devices, enabling consistent management of user interactions, motion tracking and object rendering across different hardware environments. As a result, the application can be deployed on multiple VR platforms without requiring changes to the backend code, increasing scalability and compatibility of the system.

2.3.1.3 Custom code architecture based on Dependency Injection

In order to maintain a high level of modularity and flexibility in the application, the backend uses a custom code architecture based on the **Dependency Injection** (DI) pattern. This approach allows for dynamic management of dependencies between system components, increasing the testability of the code and making it easier to be expanded. The application's structure has been designed to allow easy injection and replacement of modules without interfering with the source code of the other layers of the system, which streamlines deployment processes and reduces the risk of errors resulting from strong dependencies between components.

2.3.1.4 Vulkan Graphics API

Using the **Vulkan Graphics API** in the backend layer allows efficient management of rendering processes, optimizing the use of hardware resources and maximizing graphics performance. Vulkan offers low-level access to GPUs, enabling parallel operations and minimizing the overhead associated with real-time graphics handling. With support for asynchronous rendering operations, the system can dynamically manage the hardware load, resulting in increased smoothness of VR applications and reduced lag in displaying content.

2.3.1.5 WebGL

Integration with **WebGL** allows VR application functionality to be ported to web browsers, ensuring that content is available without the need to install additional software. WebGL allows rendering of advanced 3D scenes in a web environment, using direct access to GPU hardware acceleration. This allows the application to run on a wide range of devices, including PCs and mobile computers, making it more accessible and facilitating the distribution of training content

2.3.1.6 Vivox

The application's backend supports voice communication via **Vivox** technology, providing high-quality audio transmission in multiplayer environments. Vivox offers advanced noise cancellation algorithms, dynamic volume control and support for 3D spatial audio, resulting in a realistic representation of communication in a virtual environment. Vivox's integration with application architecture allows flexible management of voice sessions, personalization of settings and optimization of network bandwidth depending on operational conditions.

2.3.1.7 Custom server connection

The communication layer of the backend has been designed using a custom server connection management system, which features high modularity, intelligent session management and optimized data handling. This mechanism allows for dynamic resource allocation, load balancing and adaptive scaling of the server infrastructure depending on the number of users. Thanks to the intelligent data processing system, the application is able to efficiently handle real-time communications, minimizing latency and ensuring stable operation regardless of operational conditions.

2.3.1.8 ARFoundation

The application backend supports **ARFoundation** technology, enabling the development of augmented reality (AR) applications in a modular and platform-independent manner. ARFoundation is an integrated solution in the Unity ecosystem that allows easy implementation of AR features such as surface tracking, 3D object detection, image recognition and interaction with virtual elements in real space. By leveraging ARFoundation, the app gains full compatibility with popular AR platforms, including **ARKit** and **ARCore**, enabling the creation of AR experiences both on mobile devices and in VR environments. The system also supports advanced features such as depth tracking and environmental analysis, allowing for even more advanced user interaction and enhancing the user experience with augmented reality elements.

2.4 Language localisation

The application is equipped with the functionality of translation into a number of languages, including: Polish, English, Greek and Italian. The language codes are **ISO 639** compliant, ensuring universality and compatibility with international standards. These translations make the application widely available, allowing users from different regions of the world to enjoy the full functionality of the tool in the language of their choice.

Multilingual support is an essential component of global training and education, enabling seamless access to applications regardless of language barriers. To this end, the **OpenSans** font was used, which is fully compatible with all European languages and ensures adequate readability, regardless of the diacritical mark or specific alphabet. This font was chosen for its versatility and versatility in displaying texts in different languages, which allows the visual consistency of the application to be maintained.

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

Figure 17. Display UI of language selection

The process of changing the language in the application is intuitive and quick - users can change the language of the interface at any time by clicking the appropriate flag, which is a UI element accessible from a menu. After the change of language, all text elements of the application will be replaced with language versions corresponding to the newly selected language option. The mechanism is supported by downloading text elements placed on the server side at the stage of creating gameplay scenarios. As a result, all language versions of the application's text realm function within a single iteration of the system, which means there is no need to restart the application or reload content.

An added value is that the user's selected application language is stored on the server as a correlation to his or her user profile, allowing for automatic restoration of language preferences when the user logs back into the application. As a result, the user can enjoy a personalized experience without having to re-set language preferences every time the application is launched.

This approach makes the application even more accessible and useful in the context of global training deployments, eliminating language barriers and allowing easy adaptation to the needs of users from different regions and cultures. Translations into different languages, support for multiple alphabets and dynamic language switching make the application extremely flexible and ready for the needs of the international user.

2.5 Data security

Security of user data was a priority when designing the application. When a user logs in, the system does not store login data such as passwords or credentials. Instead, only authentication tokens are stored, which are used to identify the user during the session. These tokens are stored securely in memory, plus they are properly encrypted to protect them from unauthorized access. Encryption of the tokens prevents them from being read by unauthorized persons, even in the event of a breach of system security.

Communication between the application and the server is carried out exclusively via **HTTPS** (Hypertext Transfer Protocol Secure). Thanks to this, all data transfer is encrypted, which ensures the confidentiality and integrity of the transmitted information. HTTPS-level encryption ensures that user data is protected from interception by third parties, eliminating the risk of **man-in-the-middle** attacks. This type of data security ensures that the information transmitted between the application and the server is secure and cannot be decrypted by unauthorized parties.

With these protection measures, the application provides a high level of security for user data, minimizing the risk of sensitive information leakage and ensuring compliance with best practices in privacy and data security.

2.6 Multiplayer

The application supports multiplayer gameplay mode, allowing interaction and collaboration between multiple users in a virtual environment. The multiplayer logic is based on real-time synchronization of users' states, providing a seamless collaborative learning experience. Each participant in a session is represented in the VR environment by an avatar, whose position, movements and actions are continuously transmitted and synchronized with other players. By using a dedicated communication system (such as Vivox), users can have voice conversations, allowing for better collaboration and more effective group training. The possibility of real-time voice communication allows soft skills training in a virtual environment of users located in different parts of the world - a functionality almost without precedent. The server synchronizes data on positions, interactions and events in the application, ensuring that all changes are immediately visible to all participants in the gameplay session. Multiple participants can work on the same tasks, simulations or exercises, enabling effective team learning. The multiplayer logic is designed to minimize latency and ensure smooth interactions, which is especially important in a VR environment where responsiveness is critical to user comfort and efficiency.

The layer of information exchange is **WebSockets** technology. It is a communications protocol that allows two-way, full duplex, real-time communication between a client and server over a single, maintained **TCP connection**. Unlike the traditional HTTP model, which requires a new connection to

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

be established for each interaction, WebSockets allows a persistent connection to be maintained, eliminating the need for multiple requests and responses. This makes communication more efficient, faster and less resource-intensive, which is especially important in applications that require real-time data transmission, such as games and VR applications. In the context of applications, WebSockets is used to provide instant data synchronization between users. As a result, users' positions, movements and interactions are transmitted in real time to other participants in the session, ensuring a smooth and consistent experience in the virtual environment. WebSockets enables minimized latency, making real-time interactions such as voice calls or working together on tasks almost instantaneous.

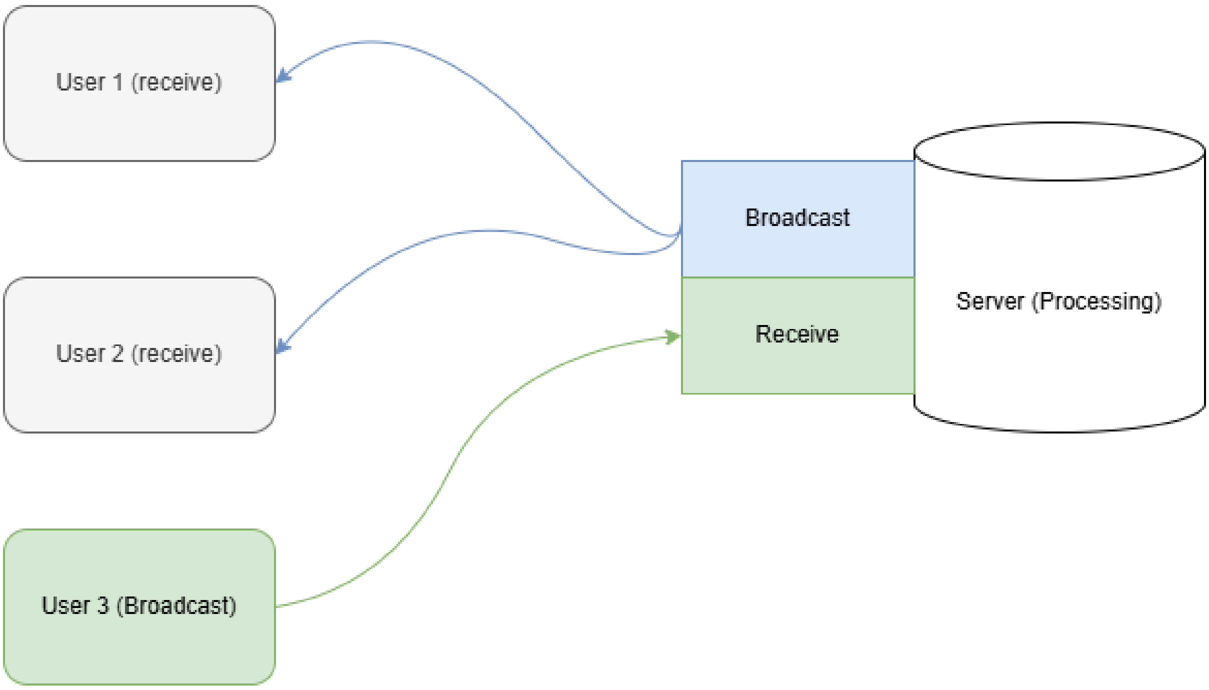

Figure 18. Diagram showing the operations supported by the server during multiplayer session

3. Methodological assumptions of the training implemented with the use of the application

3.1 Scope of content and implemented training activities

The app includes training activities from a wide range of topics. Such an approach was taken keeping in mind the goals to be served by the application (described in the initial part of the scenario). Treating the subject matter of the training layer as a collection of issues from different areas allowed us to present the audience with a very wide spectrum of possibilities for implementing training with VR tools. A kind of vertical slice through different thematic areas gives an idea of how VR technology can be used to train a wide range of competencies. At the same time, it serves as a confirmation of the technology's versatility in the context of its educational applications.

The application includes activities aimed at developing competencies in areas such as:

- Computer graphics (issues related to understanding the differences between vector and raster graphics, color theory, practical application of computer graphics for application purposes, implementation of graphic designs for specific applications, understanding the properties of graphic files of different formats, issues of resolution of graphic files),
- Website development (creation of website designs, understanding elements of websites and CMS functions, selection of website design in relation to its function and purpose, creation of mockups),
- Customer Relationship Management (issues related to the practical application of CRM systems, business context of the use of such systems),
- 3D printing (material theory, operation of 3D printers, operation of the printing process, printing parameters),
- Employer branding (theoretical issues of branding, recognition of the business context of the customer),
- Working with gifted students (examples of tools for working with gifted students using VR environments, issues of dedicated methodological approach to working with gifted students, theoretical issues of didactics).

Users of the system can acquire competencies related to the described subject areas, but the activities related to them have not been built on the basis of complete sets of learning outcomes, but rather individual learning outcomes. They have been prepared as examples of training environments utilizing VR technology (the application is, as envisioned in the proposal, a demonstration version of the technology). In addition, the described activities are a kind of background for the application's primary training function, which is soft skills training, which is implemented through the way users will interact with the simulation environment and with other system users - during multiplayer gameplay (see section: 3.2 gameplay implementation).

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

Co-funded by
the European Union

3.2 Gameplay implementation

3.2.1 Modular gameplay design

The architecture of the gameplay area and its course has been designed in a modular and segmented manner, which allows for gradual and targeted learning. The educational process is carried out within a Scenario, which is a logical training unit consisting of sequentially arranged stages. Each Scenario is divided into separate training rooms, which function as spaces dedicated to interactive knowledge acquisition.

In each room, the user has access to a variety of interactions with the educational content, which are implemented through dynamic elements of the VR environment. These interactions can include manipulating objects, analyzing visual data, using user interfaces and solving tasks in the context of the virtual environment. Upon completion of a learning stage, the user is confronted with a logic puzzle, the solution of which is a prerequisite for moving on to the next training space. This gamification mechanism provides an additional layer of interactivity, increasing engagement and the effectiveness of knowledge acquisition.

Transitions between successive training segments are implemented in an immersive manner: through the use of doors or airlocks inspired by solutions used in space stations. This mechanism reinforces the impression of smooth movement in the VR environment and provides a logical navigation structure. This approach allows for a natural separation of the various educational modules, while supporting the user's intuitive exploration of the content.

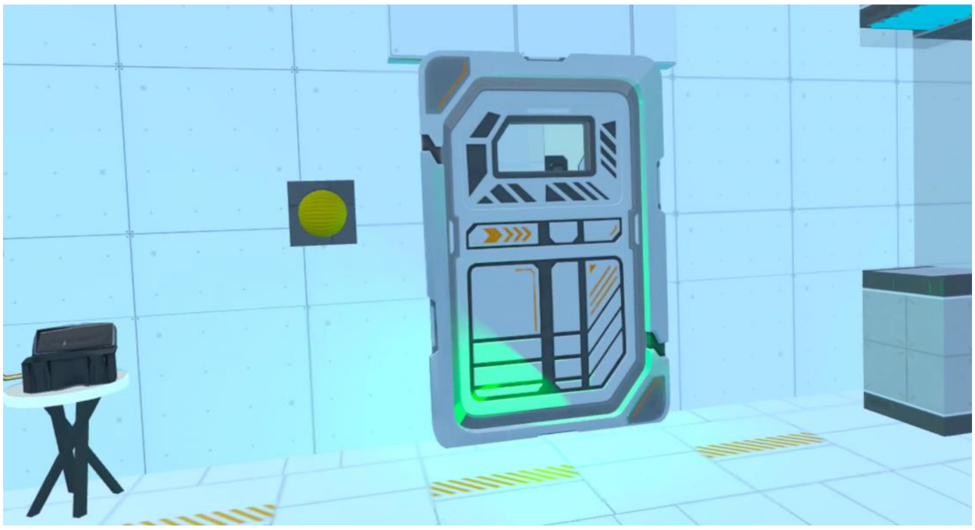

Figure 19. An example of a door separating the different segments of a gameplay scenario

The segmented structure used provides a number of technological advantages. First of all, it allows for gradual and controlled dosage of content, which reduces the effect of information overload and allows for more efficient absorption of knowledge. In addition, the modular nature of the system allows easy expansion and adaptation of Scenarios to different audiences and training topics. This allows the application to be dynamically modified and scaled depending on the needs of users and training objectives.

3.2.2 'Escape Room' concept and multiplayer gameplay

The individual rooms were built using a convention typical of so-called 'escape rooms'. Users get into the gameplay space (segments described earlier) and have to perform tasks, solve problems and overcome obstacles, built using learning activities (described in section 3.1 Scope of content and implemented training activities). An task could be, for example, to print a key on a 3D printer, using the appropriate material. In this way, users are forced to utilize competencies related to each thematic area, in order to achieve gameplay progress. Users are not allowed to leave the room (and move on to the next segment of the scenario) until they have solved all the tasks distributed in the room they are currently in.

The use of such a formula allows to maximize the involvement of users, with simultaneous interpretation and practical use of competencies from different thematic areas.

As mentioned earlier, a key aspect of the training scenarios is the implementation of soft skills training. Individual gameplay segments are designed to require users to cooperate (e.g., tasks that cannot be completed alone). Effectively, in order to complete a scenario, users must communicate effectively with each other, cooperate as a team and manage their time appropriately. Activities of this type fully utilize the functionalities associated with multiplayer gameplay mode and are designed to facilitate communication and teamwork.

Figure 20. An example of multiplayer gameplay - an activity that requires cooperation

3.2.3 Examples of activities implemented in the application

Below are examples of selected activities implemented as part of the gameplay flow. This is not a complete list of all utilized activities, but rather an exemplary representation of the graphical style and implementation of activities in the simulation environment. Some elements of the environment can be used repeatedly and do not represent a single instance of interaction (e.g., multiple items can be printed on a 3D printer, changing thermoplastic materials, etc.).

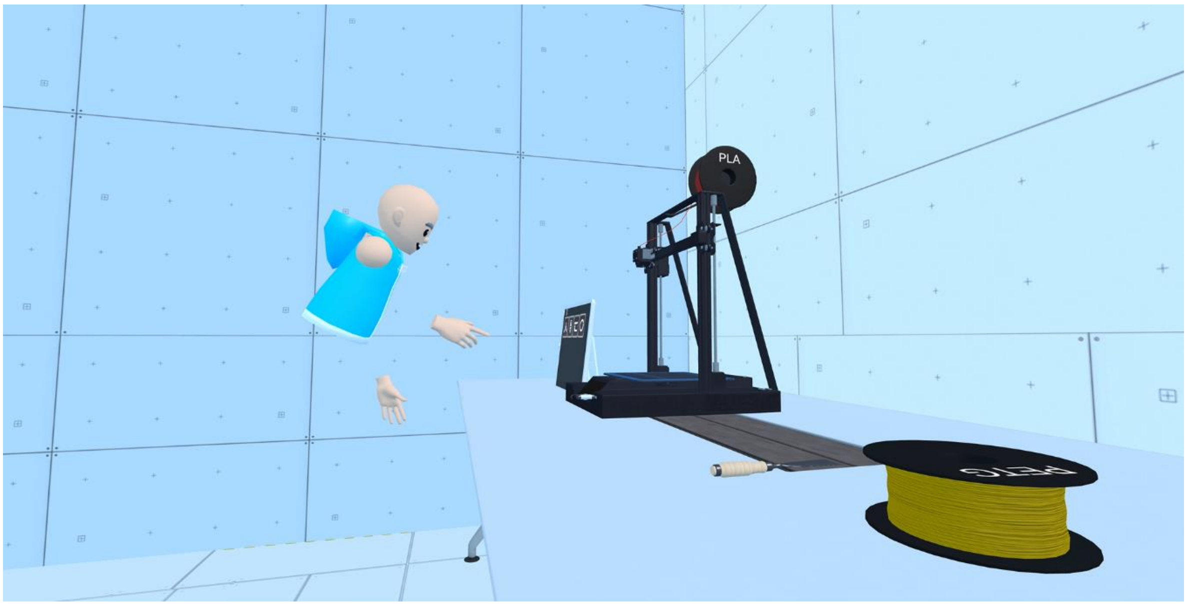

Figure 21. An example of a 3D printing activity - the user selects a model to print, having previously selected the material and set the printing parameters

Figure 22. Gallery with an exposition of information on color theory and the cultural significance of colors

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

Co-funded by
the European Union

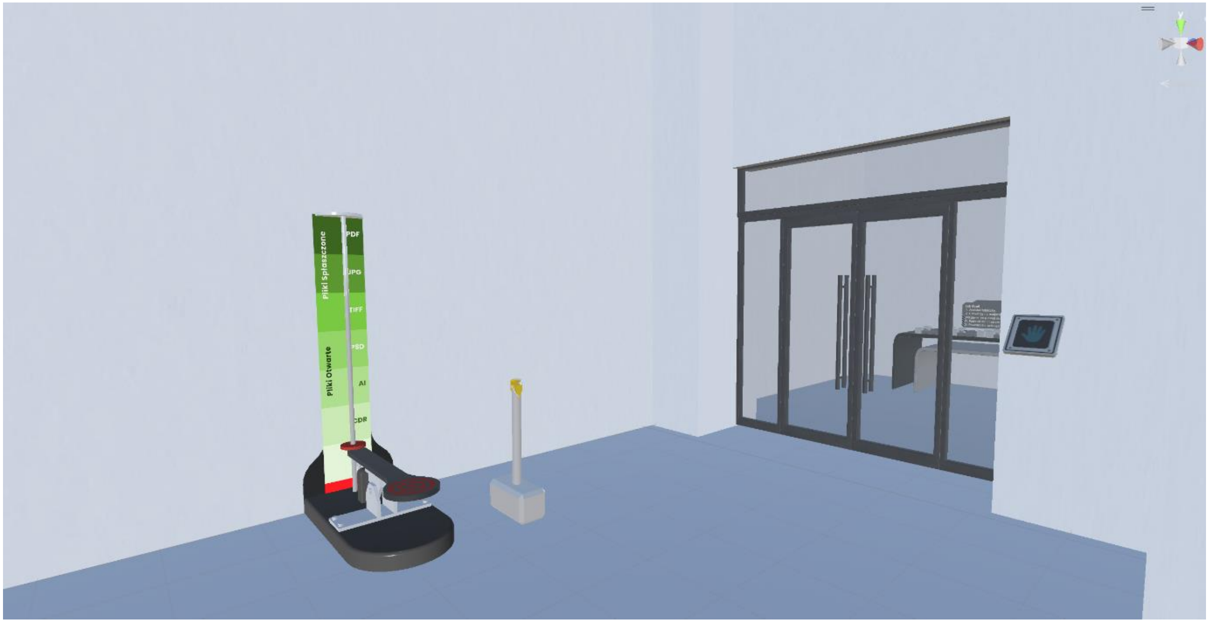

Figure 23. Activity related to the issue of file formats - interaction involves hitting items placed on the platform with a hammer to produce an effect

Figure 24. An example of a position with exposure of information that will be required for another activity - one of the strengths of VR technology, allowing not only the presentation of information, but also the visualization of data in the form of objects

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them."

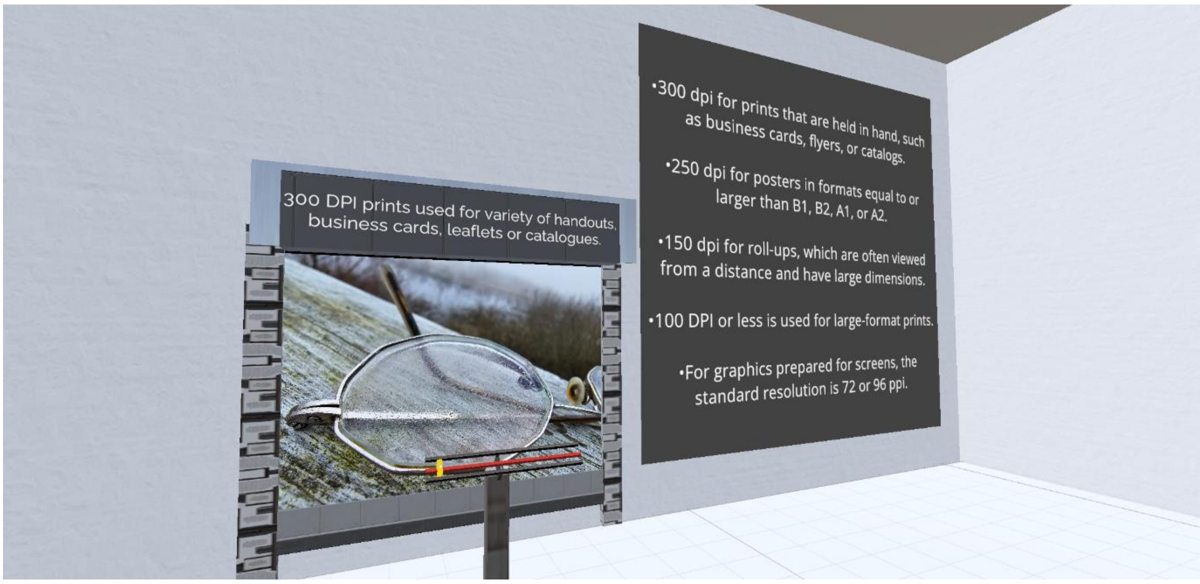

Figure 25. Activity to inform about the use of different DPI levels, along with an interactive simulation

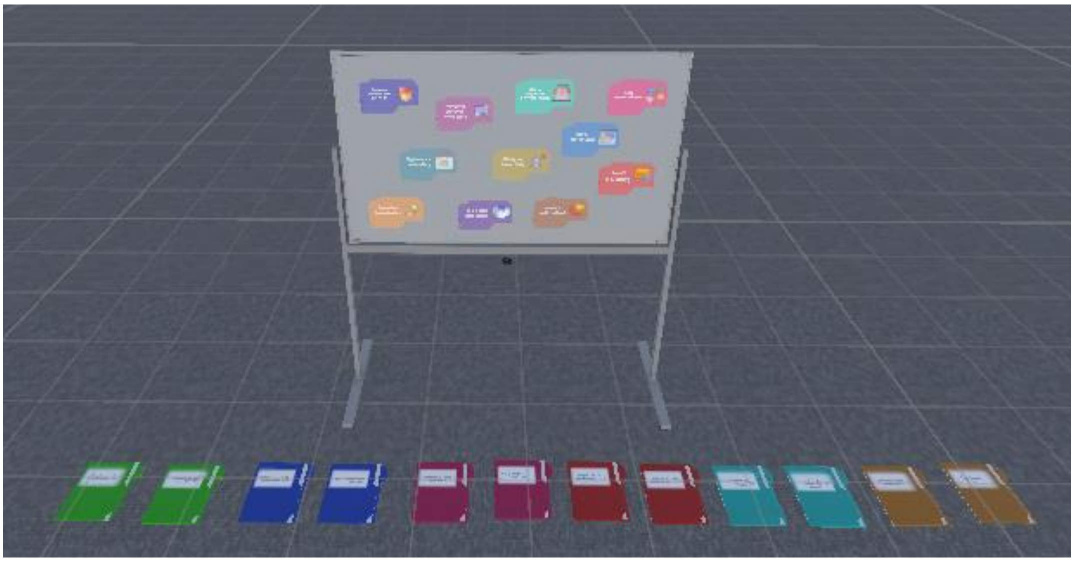

Figure 26. An example of a customer relationship management task - folders with data are hidden in different parts of the gameplay space and contain the information needed to complete the exercise. A good example of the application of the 'escape room' convention in an educational activity.

Co-funded by
the European Union

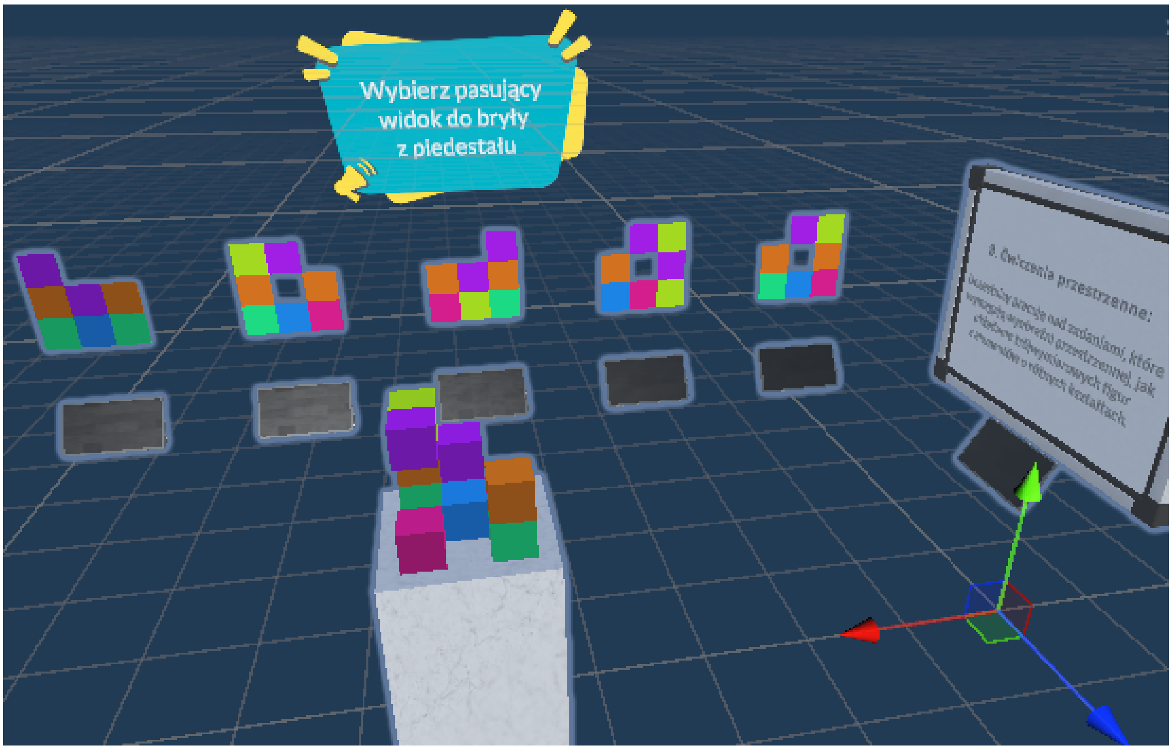

Figure 27. An example of an activity on working with gifted students - VR environments are phenomenally successful as an educational tool used in special needs teaching

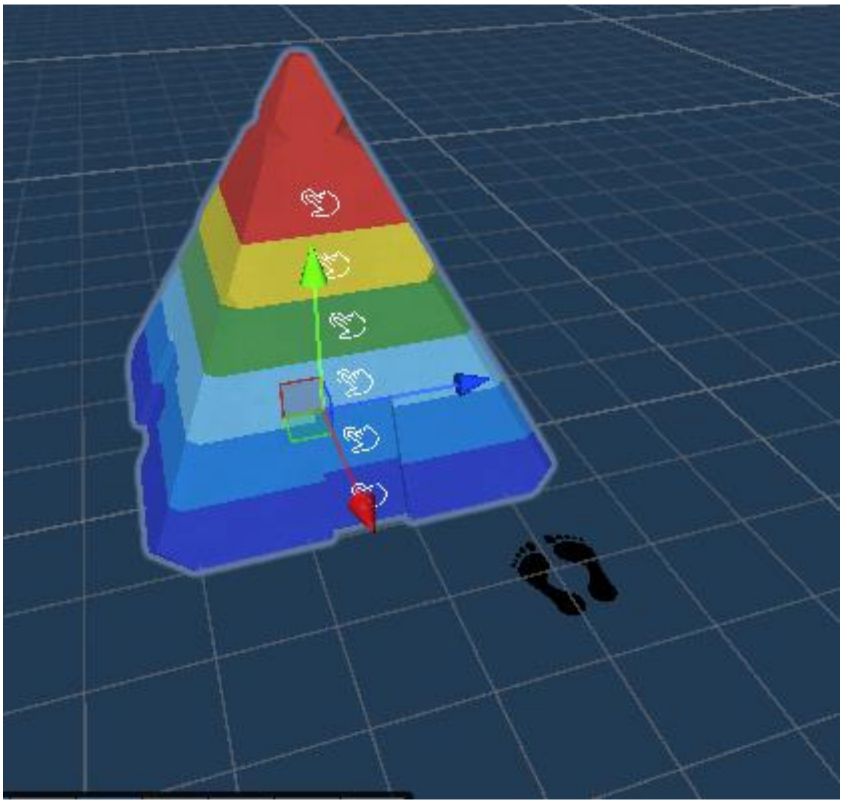

Figure 28. Activity related to the well-recognized pyramid of needs

“Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor the granting authority can be held responsible for them.”

Co-funded by
the European Union

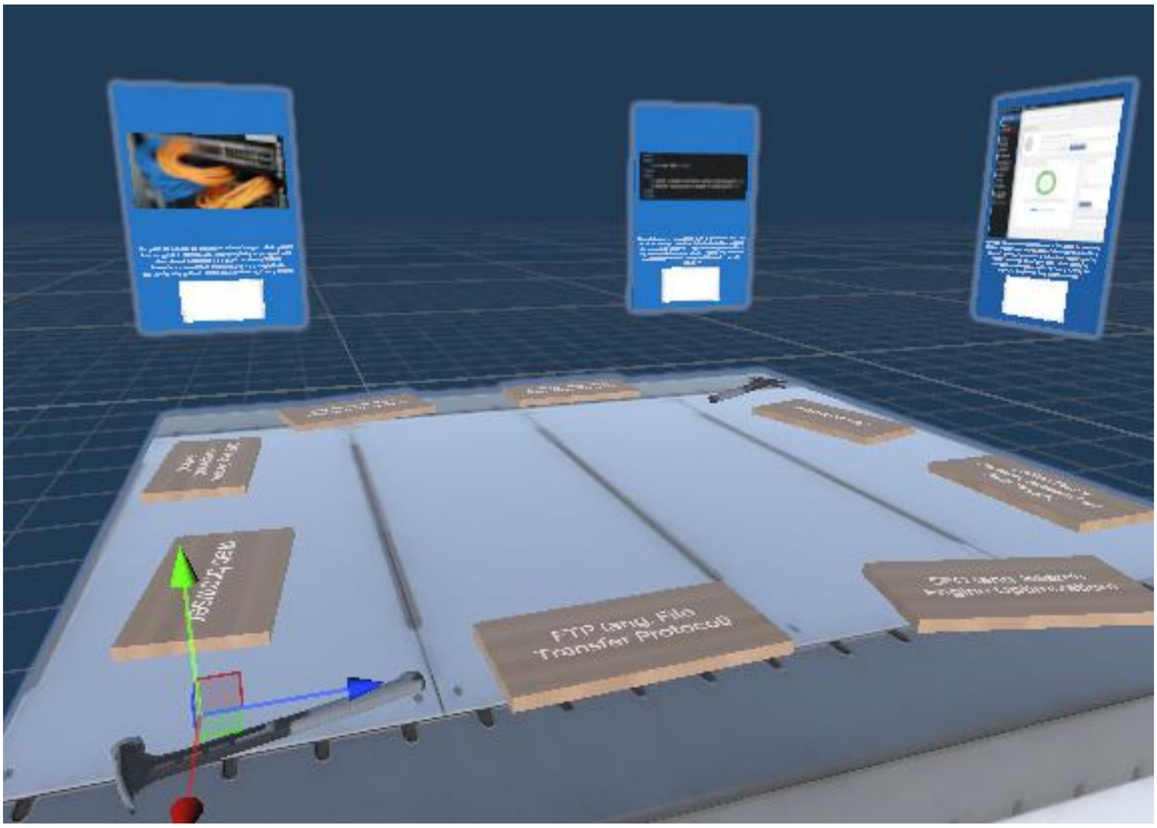

Figure 29. An interactive version of the matching activity - the user must pair a definition with a description and visualization by nailing a plaque with its phrasing in correct spots