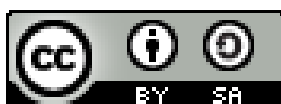


## Zadanie O2/A6

# WDROŻENIE ULEPSZEŃ - TECHNICZNE ULEPSZENIE NARZĘDZIA VIRTUAL REALITY (VR)



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/)



"The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein."

## INDEX

|  |                                      |
|--|--------------------------------------|
| 1. INTRODUCTION .....  | <b>¡Error! Marcador no definido.</b> |
| 1.1. Methodology and operation of the application.....   | 4                                    |
| 2. DEVELOPMENT OF THE TOOL. ....   | 4                                    |
| 3. SCENE DESIGN. ....  | 5                                    |
| 3.1. Scene selection.....  | 5                                    |
| 3.1.1. Drones (Unmanned Aerial Vehicles) – Preparing for flights on construction sites in daylight.....        | <b>¡Error! Marcador no definido.</b> |
| 3.1.2. Drones (Unmanned Aerial Vehicles) – Flying on construction sites in favourable weather conditions ..... | <b>¡Error! Marcador no definido.</b> |
| 3.1.3. Drones (Unmanned Aerial Vehicles) – Flying on construction sites in adverse weather conditions .....    | <b>¡Error! Marcador no definido.</b> |
| 3.1.4. Drones (Unmanned Aerial Vehicle) – Preparing for flights on construction sites at night                 | <b>¡Error! Marcador no definido.</b> |
| 3.1.5. Autonomous Site Transport Vehicle – Indoor site conditions .....  | 7                                    |
| 3.1.6. Autonomous Site Transport Vehicle – External and outdoor site conditions                                | <b>¡Error! Marcador no definido.</b> |
| 3.1.7. Remote Controlled Equipment (Demolition Robots) – General Requirements                                  | <b>¡Error! Marcador no definido.</b> |
| 3.1.8. Remote Controlled Equipment (Demolition Robots) – Indoor site conditions                                | <b>¡Error! Marcador no definido.</b> |
| 3.1.9. Remote Controlled Equipment (Demolition Robots) – External and outdoor site conditions.....             | <b>¡Error! Marcador no definido.</b> |
| 3.1.10. Remote Controlled Equipment (Diggers/excavators) - External and outdoor site conditions.....           | <b>¡Error! Marcador no definido.</b> |
| 3.2. General instructions.....   | <b>¡Error! Marcador no definido.</b> |
| 3.3. Fail Panel .....  | 10                                   |
| 3.4. Success Panel .....   | 10                                   |
| 3.5. Task (Example) .....  | 10                                   |
| 4. Development of the 3D Virtual Environment .....   | <b>¡Error! Marcador no definido.</b> |
| 5. FUNCTIONALITY DEVELOPMENT AND IMMERSIVE EXPERIENCE. ....  | 17                                   |
| 5.1. Design and architecture .....   | 17                                   |
| 5.1.1. Main Menu .....   | 17                                   |
| 5.2. Data modelling .....  | 19                                   |
| 5.3. SDK Integration .....   | 19                                   |

|        |  |                                      |
|--------|--|--------------------------------------|
| 5.4.   | Event management .....                     | 20                                   |
| 5.4.1. | Prepare the scene for VR interaction ..... | 20                                   |
| 5.4.2. | Controller input .....                     | <b>¡Error! Marcador no definido.</b> |
| 5.4.3. | Quiz Manager .....                         | 23                                   |
| 5.4.4. | Translator .....                           | 24                                   |
| 5.4.5. | Interaction with rays .....                | 24                                   |
| 6.     | Conclusion .....                           | <b>¡Error! Marcador no definido.</b> |

## 1. WPROWADZENIE

Konsorcjum SafeCROBOT opracowało wciągające i interaktywne narzędzie szkoleniowe oparte na rzeczywistości wirtualnej, które składa się z 10 scenariuszy zagrożeń (każdy scenariusz obejmuje jeden typ robota budowlanego/autonomicznego pojazdu i związane z nim zagrożenia) oraz środków zapobiegawczych niezbędnych do ich złagodzenia. Głównym celem immersyjnego narzędzia opartego na wirtualnej rzeczywistości opracowanego w ramach tego projektu będzie wykorzystanie go do szkolenia pracowników budowlanych, aby umożliwić im bezpieczniejszą pracę obok robotów budowlanych i autonomicznych urządzeń na placach budowy.

Immersyjne narzędzie szkoleniowe zostało udoskonalone, dlatego też wersja beta narzędzia została zmodyfikowana o ulepszenia zidentyfikowane podczas ewaluacji przez współpracowników, a także testowania na kursach uczestników. Dodatkowo uwzględniono wnioski techniczne z Seminariów Międzynarodowych.

### 1.1. Metodologia i działanie aplikacji

Badania i rozwój SafeCROBOT zostały podzielone na kilka odrębnych części odpowiadających rozwojowi zaplanowanych pakietów prac.

W kolejnych częściach opisano metodologię ewaluacji, na której oparto projekt, a także jej działanie i wdrożone usprawnienia.

## 2. OPRACOWANIE NARZĘDZIA

SafeCROBOT (zwany dalej "narzędziem") bazuje na immersyjnej rzeczywistości wirtualnej. Obecnie najpowszechniejszym sposobem tworzenia aplikacji w tej technologii są silniki do tworzenia gier, takie jak Unity3D.

Rozwój narzędzia można podzielić na trzy główne zadania, które zostaną szeroko omówione w kolejnych rozdziałach:

- **Projektowanie scen.**
- **Rozwój środowisk 3D.**
- **Rozwój funkcjonalności i immersyjnych doświadczeń.**

W celu sprawnego i efektywnego przeprowadzenia rozwoju należy zauważyć, że wykorzystano różne technologie:

- **Autodesk Revit:** Autodesk Revit jest oprogramowaniem do modelowania informacji o budynku przeznaczonym dla architektów, inżynierów strukturalnych, mechanicznych, elektrycznych i hydraulicznych, projektantów i wykonawców.

- **Blender:** Wieloplatformowe oprogramowanie dedykowane do przetwarzania, modelowania, renderowania i innych zadań związanych z grafiką 3D. Jest open source i ma dużą społeczność z filmami i dokumentacją, co sprawia, że bardzo łatwo się go nauczyć.
- **Unity3D:** To kolejne oprogramowanie open-source, silnik do tworzenia gier, który wspomógł rozwój wciągającego środowiska wirtualnego. Jest wysoce skalowalny, umożliwiając użytkownikom dodawanie modułów za pomocą skryptów kodu, a także ma ciekawą społeczność i dokumentację.
- **Oculus Quest:** Jest to bardzo przystępne i dostępne urządzenie Virtual Reality. Opracowane przez Metę jako jeden z filarów jej nowej filozofii. Nie wymagają kabli do użycia i mają łatwą integrację z Unity.

### 3. PROJEKTOWANIE SCEN

Jest to proces, w którym wymyślane są poszczególne kroki, jakie należy wykonać od momentu wejścia do narzędzia do momentu zakończenia zanurzenia. Innymi słowy, generowanie scenariusza rozwoju aplikacji.

Pierwszym krokiem była identyfikacja różnego rodzaju zagrożeń związanych z zastosowaniem robotów/autonomicznych pojazdów na placach budowy poprzez rygorystyczny przegląd literatury. Po zidentyfikowaniu i skategoryzowaniu zagrożeń (zagrożenie dla siebie i zagrożenie dla innych); zamodelowano 10 scenariuszy ryzyka opartych na zidentyfikowanych zagrożeniach przy użyciu wyżej wymienionych narzędzi. Wybrane scenariusze są następujące:

#### 3.1. Wybór scen

##### 3.1.1. Drony (Bezzałogowe Statki Powietrzne) - przygotowanie do lotów na placach budowy w trakcie dnia

Podczas tego scenariusza użytkownik wcieli się w rolę pilota bezzałogowego statku powietrznego (BSP). Zadaniem użytkownika będzie odpowiednie przygotowanie się do lotu w ciągu dnia na placu budowy. Użytkownik znajduje się w biurze budowy. W kontenerze znajduje się następujący sprzęt: dron, komplet naładowanych baterii, tablet oraz dokumentacja budowy. Na ścianie użytkownik znajdzie zagospodarowanie terenu budowy. Biorąc pod uwagę kilka najważniejszych kwestii związanych z bezpieczeństwem latania dronami na placach budowy

użytkownicy są następnie zobowiązani do odpowiedzi na pytania w quizie znajdującym się w obrębie sceny.

### 3.1.2. Drony (Bezzałogowe Statki Powietrzne) - Latanie na placach budowy w sprzyjających warunkach atmosferycznych

Podczas tego scenariusza użytkownik wcieli się w rolę pilota bezzałogowego statku powietrznego (BSP). Zadaniem będzie wykonanie misji (nalot na plac budowy) z wykorzystaniem drona w pogodny i słoneczny dzień. Podczas nalotu będą musieli zwracać uwagę na otoczenie oraz komunikaty pojawiające się na ekranie kontrolera dotyczące parametrów technicznych lotu. Biorąc pod uwagę kilka najważniejszych kwestii związanych z bezpieczeństwem latania dronami na placach budowy użytkownicy są następnie zobowiązani do odpowiedzi na pytania w quizie w ramach scenki.

### 3.1.3. Drony (Bezzałogowe Statki Powietrzne) - Latanie na placach budowy w niesprzyjających warunkach atmosferycznych

Podczas tego scenariusza użytkownicy wcielą się w rolę pilota bezzałogowego statku powietrznego (BSP). Zadaniem będzie wykonanie misji (nalot na plac budowy) z wykorzystaniem drona podczas niekorzystnych warunków atmosferycznych, w tym wiatru, oraz deszczu. Podczas nalotu użytkownicy będą musieli zwracać uwagę na otoczenie oraz komunikaty pojawiające się na ekranie kontrolera dotyczące parametrów technicznych lotu. Biorąc pod uwagę kilka najważniejszych kwestii związanych z bezpieczeństwem latania dronami na placach budowy użytkownicy są następnie zobowiązani do udzielenia odpowiedzi na pytania w quizie znajdującym się w obrębie sceny.

### 3.1.4. Drony (Bezzałogowe Statki Powietrzne) - przygotowanie do lotów na placach budowy w nocy

Podczas tego scenariusza użytkownicy wcielą się w rolę pilota bezzałogowego statku powietrznego (BSP). Zadaniem będzie odpowiednie przygotowanie się do lotu w nocy na placu budowy. Użytkownik znajduje się w biurze budowy. W kontenerze użytkownik znajdzie następujący sprzęt: dron, komplet naładowanych baterii, tablet, dokumentację budowy oraz dodatkowe oświetlenie. Rozejrzyj się po biurze i przygotuj się do lotu. Biorąc pod uwagę kilka najważniejszych kwestii związanych z bezpieczeństwem latania dronami na placach budowy użytkownicy są następnie zobowiązani do udzielenia odpowiedzi na pytania w quizie znajdującym się w obrębie sceny.

### 3.1.5. Autonomiczny pojazd transportu naziemnego - warunki wewnętrzne terenu

Moduł ten wprowadza użytkowników w wymagania BHP dotyczące obsługi Autonomicznych Pojazdów Transportowych (ATV) w warunkach budowy wewnątrz budynków. ATV to pojazd zdolny do samodzielnego działania. Są one zazwyczaj kołowe lub gąsienicowe i różnią się wielkością. Do transportu wewnątrz budynków ATV są zwykle małych i średnich rozmiarów. Przyjmij, że ATV używany w tym module jest elektryczny i ma następujące wymiary: L-1200mm, H-600mm i W-700mm i może przewozić ładunki do 500kg. Użytkownicy są zobowiązani do P obserwowania środowiska i sprzętu w celu zidentyfikowania wszelkich kwestii, które uważają za zagrożenia dla zdrowia i bezpieczeństwa. Rozważając niektóre z najważniejszych kwestii bezpieczeństwa w zarządzaniu ATV na placu budowy, użytkownicy są następnie zobowiązani do odpowiedzi na pytania w quizie w ramach sceny.

### 3.1.6. Autonomiczny pojazd transportu terenowego - warunki zewnętrzne i terenowe

Moduł ten wprowadza użytkownika w wymagania dotyczące bezpieczeństwa i higieny pracy przy obsłudze autonomicznych pojazdów transportowych (ATV) w warunkach zewnętrznych lub zewnętrznych placów budowy. W przypadku działań na placu budowy lub w terenie zewnętrznym pojazdy ATV różnią się od małych do bardzo dużych urządzeń. ATV używany w tej scenie to duże wozidło używane do transportu materiałów na placu budowy. Typowe materiały to kruszywo, materiały z wykopów lub rozbiórki. Obserwuj środowisko i sprzęt, aby zidentyfikować wszelkie kwestie, które uważasz za zagrożenia dla zdrowia i bezpieczeństwa. Rozważając niektóre z najważniejszych kwestii bezpieczeństwa przy kierowaniu pojazdem ATV na placu budowy, użytkownicy muszą następnie odpowiedzieć na pytania w quizie znajdującym się w tej scenie.

### 3.1.7. Urządzenia zdalnie sterowane (roboty burzące) - wymagania ogólne

Moduł ten zapoznaje użytkowników z wymogami BHP dotyczącymi obsługi zdalnie sterowanego robota wyburzeniowego w warunkach placu budowy (wewnątrz i na zewnątrz). Wykonawca Smith zleca dwóm swoim pracownikom, Marcowi i Gordonowi, wyburzenie kilku ścian wewnątrz/zewnątrz dużej fabryki przemysłowej. Ma być użyty zdalnie sterowany robot wyburzeniowy. Młot hydrauliczny został już zamontowany. Marc i Gordon nigdy wcześniej nie pracowali z robotem wyburzeniowym, ale nie mogą się doczekać. Proszę dołączyć do Marca i Gordona w ich pracy i dowiedzieć się o zagrożeniach dla zdrowia i bezpieczeństwa podczas pracy



z robotem wyburzeniowym. Następnie użytkownicy sprawdzą i utrwalą swoją wiedzę, rozwiązując quiz.

#### 3.1.8. Urządzenia zdalnie sterowane (roboty wyburzeniowe) - Warunki w pomieszczeniach zamkniętych

Moduł ten zapoznaje użytkowników z wymogami BHP dotyczącymi obsługi zdalnie sterowanego robota wyburzeniowego w warunkach placu budowy (wewnątrz i na zewnątrz). Wykonawca Smith zleca dwóm swoim pracownikom, Marcowi i Gordonowi, wyburzenie kilku ścian wewnątrz/zewnątrz dużej fabryki przemysłowej. Ma być użyty zdalnie sterowany robot wyburzeniowy. Młot hydrauliczny został już zamontowany. Marc i Gordon nigdy wcześniej nie pracowali z robotem wyburzeniowym, ale nie mogą się doczekać. Proszę dołączyć do Marca i Gordona w ich pracy i dowiedzieć się o zagrożeniach dla zdrowia i bezpieczeństwa podczas pracy z robotem wyburzeniowym. Następnie użytkownicy sprawdzą i utrwalą swoją wiedzę, rozwiązując quiz.

#### 3.1.9. 3.1.9. Urządzenia zdalnie sterowane (roboty rozbiórkowe) - Warunki zewnętrzne i terenowe

Moduł ten zapoznaje użytkowników z wymogami BHP dotyczącymi obsługi zdalnie sterowanego robota wyburzeniowego w warunkach placu budowy (wewnątrz i na zewnątrz). Wykonawca Smith zleca dwóm swoim pracownikom, Marcowi i Gordonowi, wyburzenie kilku ścian wewnątrz/zewnątrz dużej fabryki przemysłowej. Ma być użyty zdalnie sterowany robot wyburzeniowy. Młot hydrauliczny został już zamontowany. Marc i Gordon nigdy wcześniej nie pracowali z robotem wyburzeniowym, ale nie mogą się doczekać. Proszę dołączyć do Marca i Gordona w ich pracy i dowiedzieć się o zagrożeniach dla zdrowia i bezpieczeństwa podczas pracy z robotem wyburzeniowym. Następnie użytkownicy sprawdzą i utrwalą swoją wiedzę, rozwiązując quiz.

#### 3.1.10. Urządzenia zdalnie sterowane (koparki) - Warunki zewnętrzne i terenowe

Moduł ten wprowadza użytkowników w wymagania dotyczące bezpieczeństwa i higieny pracy przy obsłudze zdalnie sterowanego sprzętu na przykładzie koparek. Scenariusz przedstawia budowę w warunkach zewnętrznych lub na wolnym powietrzu. Koparka lub kopacz to sprzęt składający się z wysięgnika, łyżki i kabiny na obrotowej platformie. Służą one przede wszystkim do kopania i wydobywania materiału. Zazwyczaj są to urządzenia kołowe lub gąsienicowe i różnią się wielkością. Koparka użyta w tej scenie jest średniej wielkości. Użytkownicy są zobowiązani do obserwowania środowiska i sprzętu, aby zidentyfikować wszelkie kwestie, które uważają za zagrożenia dla zdrowia i bezpieczeństwa. Rozważając niektóre z najważniejszych



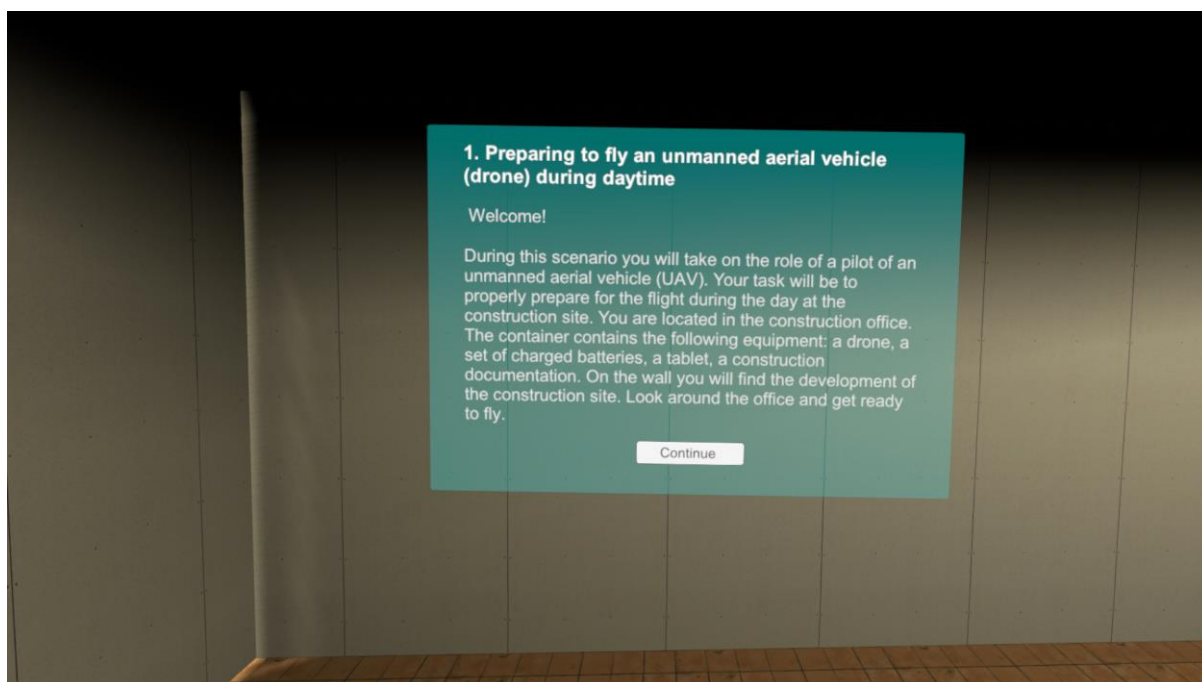
kwestii bezpieczeństwa związanych z zarządzaniem zdalnie sterowaną koparką na placu budowy, użytkownik musi odpowiedzieć na pytania w quizie dotyczącym tej sceny.

W oparciu o powyższy scenariusz sceny, modelowanie 3D wirtualnego środowiska budowlanego zostało przeprowadzone przy użyciu wcześniej szczegółowo opisanych narzędzi do modelowania.

Aby zasymulować każdy z etapów podczas gdy użytkownicy znajdują się w scenie, wygenerowano różne panele z informacjami, które użytkownik będzie musiał przeczytać, aby działać. Panele znajdujące się w każdej scenie można podzielić na następujące typy.

### 3.2. Instrukcje ogólne

Są to panele, które pojawiają się na początku każdej sceny. Wyjaśniają one procedurę, którą należy wykonać, aby dotrzeć do misji, a także ogólne informacje na temat niektórych narzędzi. Każda misja rozpocznie się od panelu powitalnego i dwóch paneli z podstawowymi instrukcjami.



Rysunek 1. Ekran powitalny z informacjami i instrukcjami dotyczącymi sceny

Panel Quizu: W tym panelu zostanie zaprezentowany zestaw pytań związanych z zagrożeniami, powiązanych z odpowiednią sceną, a użytkownicy będą mogli użyć wskaźnika laserowego i przycisku spustowego na kontrolerze, aby odpowiedzieć na pytania, które zostaną zarejestrowane, a wyniki zostaną wyświetlone na końcu quizu.

### 3.3. Panel błędu

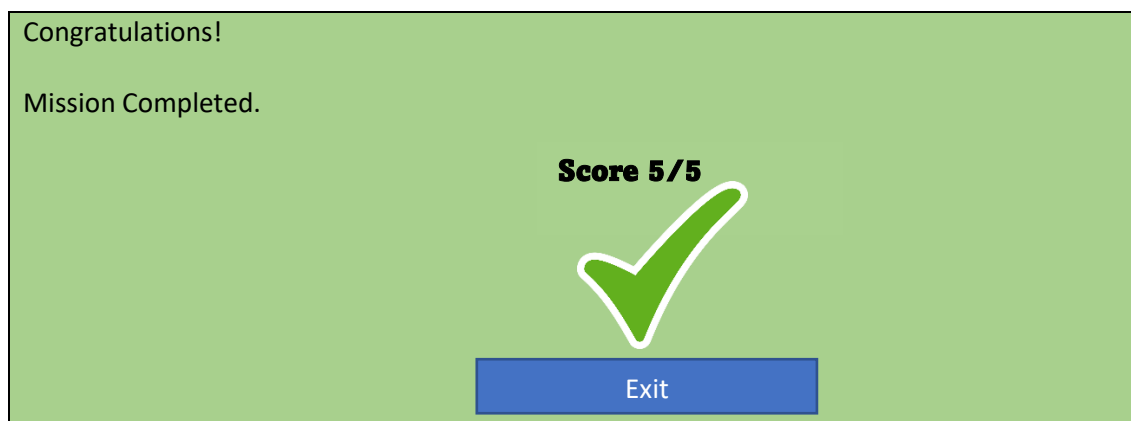
Za każdym razem, gdy misja zakończy się niepowodzeniem, pojawi się następujący panel:



Rysunek 2. Panel błędu.

### 3.4. Panel sukcesu

Za każdym razem, gdy misja zostanie ukończona prawidłowo, pojawi się następujący panel:



Rysunek 3. Paanel sukcesu.

### 3.5. Zadanie (Przykład)

Panele te będą się różnić w zależności od wykonywanego zadania, zawierając informacje, pytania, dialogi lub polecenia do wykonania.

#### Zadanie 5:

Autonomiczny Pojazd Transportu Terenowego - warunki terenowe w pomieszczeniach zamkniętych (proszę zapoznać się ze szczegółami w punkcie 3):

Continue

|  |                 |
|--|-----------------|
| <b>Quiz:</b><br>Które z poniższych zagrożeń dla zdrowia istnieje na terenie, który właśnie przejrzałeś?<br>1. Kolizja (v)<br>2. Zaburzenia mięśniowo-szkieletowe<br>3. Uwięzienie (v)<br>4. Zgniecenie (v)<br>5. Stłuczenie  |                 |
| <b>Quiz</b><br>Które z poniższych stwierdzeń dotyczących sceny jest prawdziwe?<br>1. Planowanie trasy dla pojazdu było złe (v)<br>2. Systemy ostrzegawcze pojazdu były bardzo dobre<br>3. Znaki ostrzegawcze i bezpieczeństwa są nieodpowiednie (v)<br>4. Istnieje ryzyko przewrócenia się pojazdu<br>5. Operator lub bankier nie jest możliwy do zidentyfikowania (v)<br>6. Pracownicy zachowali odpowiednią odległość między sobą a pojazdem<br>7. Pojazd musi być oddzielony od pracowników<br>8. Słabe rozgraniczenie przejść dla pieszych (v) |                 |
| <b>Quiz</b><br>Która z poniższych czynności może przyczynić się do powstania ryzyka upadku maszyny?<br>1. Trasa pojazdu zbyt blisko otworów (v)<br>2. Miejsce pracy jest zbyt małe dla pojazdu<br>3. Niezrównoważony ładunek i przeciążenie (v)<br>4. Brak zastosowania barykad<br>5. Hałas  |                 |
| (Panel sukcesu)  | (Panel porażki) |

Tabela 1. Przykładowe zadania quizu

## 4. Opracowanie wirtualnego środowiska 3D

Jako silnik gry wykorzystano Unity 3D, a modele 3D opracowane na potrzeby tworzenia scen nazywane są obiektami gry. Tym samym drugie główne zadanie polega na wygenerowaniu każdego z obiektów lub aktywów, które zostały bezpośrednio i pośrednio zdefiniowane w skryptach wygenerowanych w poprzednim punkcie.

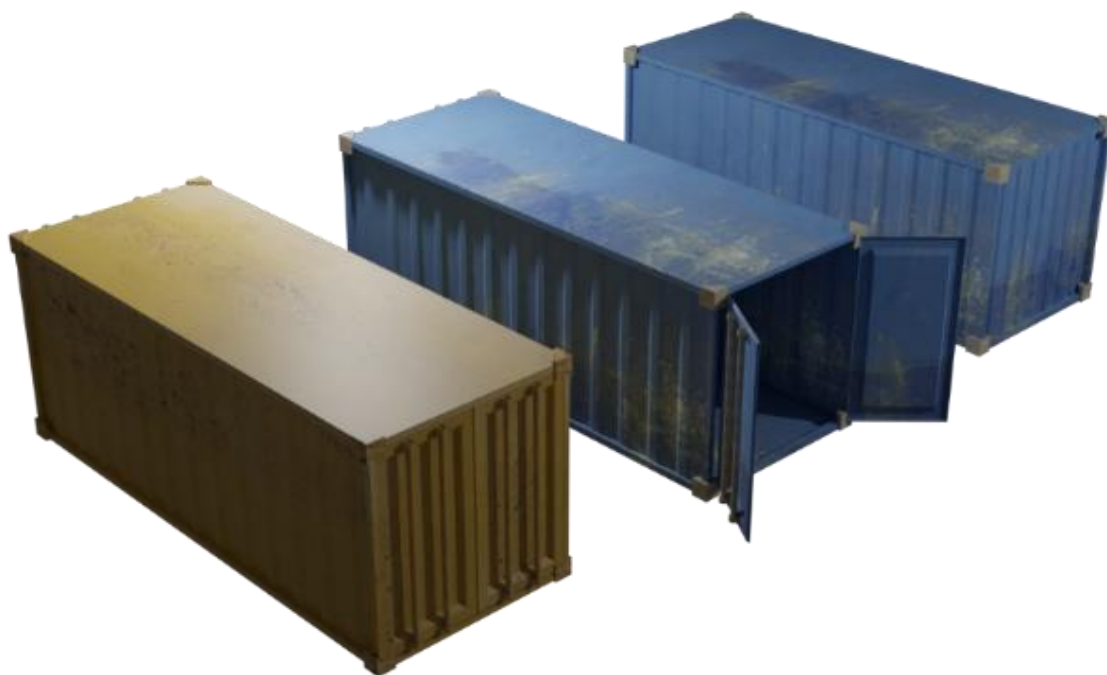
Proces ten jest z kolei podzielony na trzy zadania:

- **Modelowanie:** Matematyczna reprezentacja trójwymiarowego obiektu jest wykonywana przy użyciu określonego oprogramowania. W tym przypadku wykorzystano Blender i Autodesk Revit. Niektóre obiekty gry zostały pozyskane z magazynu zasobów i są gotowe do użycia.

- **Tworzenie:** Jest to proces, w którym poszczególne części obiektu są rozdzielane w celu stworzenia sterowalnego systemu, który umożliwia efektywne i wydajne animacje. Wykorzystano do tego samą funkcjonalność animacji Unity3D.
- **Teksturowanie:** Jest to proces, dzięki któremu obiekty 3D otrzymują kolor i szczegóły. Do tego celu wykorzystany został Blender oraz wbudowany w Unity edytor materiałów.

Lista obiektów opracowanych na potrzeby tego projektu jest bardzo szeroka i różnorodna, od bardzo prostych obiektów po skomplikowane maszyny z dużą ilością szczegółów. Wygenerowane obiekty można podzielić na następujące kategorie.

- Kontenery, Budynki, Sklepy, humanoidy i inne elementy takie jak (nie ograniczone do) kurz, deszcz itp. aby odtworzyć realistyczne środowisko budowy.



Rysunek 4. Kontenery (Przykładowe obiekty gry używane do tworzenia scen)



Rysunek 5. Robot wyburzeniowy (przykładowy obiekt użyty do tworzenia scen)



Rysunek 6. Robotnicy budowlani (przykładowy obiekt użyty do stworzenia sceny)





Rysunek 7. Robot wyburzeniowy w trakcie sceny

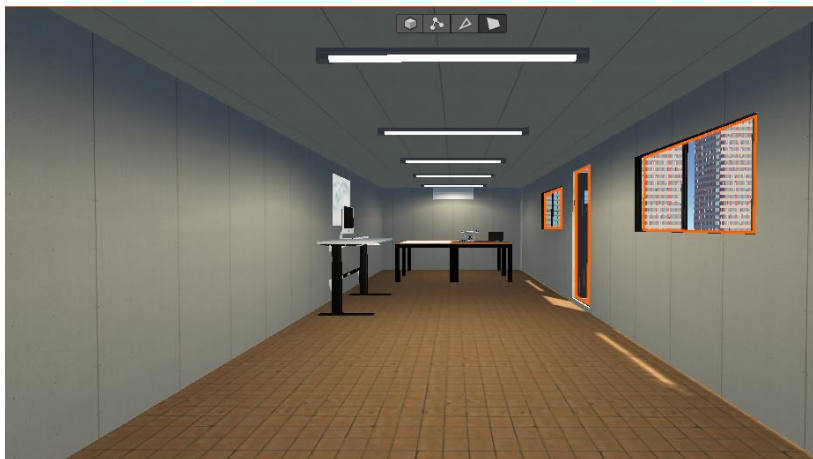


Rysunek 8. Aktywny plac budowy (Opracowany przy użyciu obiektów gry)

Po opracowaniu każdego z obiektów do wykorzystania przygotowywane są scenariusze, tak aby opracowanie funkcjonalności było jak najdokładniejsze i jak najbardziej zbliżone do sytuacji końcowej. Proces ten nie jest zazwyczaj ostateczny, gdyż w procesie testowania mogą pojawić się modyfikacje. Proces ten został przeprowadzony z wykorzystaniem silnika graficznego Unity, gdyż to właśnie on zostanie wykorzystany do wygenerowania aplikacji.

Jeśli przeanalizujemy opisy każdej z opisanych w poprzednim punkcie sytuacji, to zobaczymy, że mamy do czynienia z trzema różnymi scenariuszami:

- **Kontener** (biuro budowy): W kontenerze użytkownik znajdzie następujący sprzęt: dron, komplet naładowanych baterii, tablet, dokumentację budowy oraz dodatkowe oświetlenie.



Rysunek 9. Kontener biurowy



Rysunek 10. Dron i akcesoria w biurze kontenerowym

- **Plac budowy:** Ten model przedstawia plac budowy z, rusztowaniami, szkieletem budynku i odpowiednimi pojazdami autonomicznymi.





Rysunek 11. Plac budowy

## 5. ROZWÓJ FUNKCJONALNOŚCI I IMMERSYJNE DOŚWIADCZENIE

Rozwój funkcjonalności to proces, w którym cała zebrana teoria nabiera kształtu. Innymi słowy, celem tego zadania jest uczynienie narzędzia użytecznym. Do rozwoju wykorzystano silnik graficzny Unity, ponieważ zapewnia on prostą integrację z Wirtualną Rzeczywistością i posiada dużą społeczność.

Projekty w Unity są podzielone na Sceny, które mogą być przeplatane zdarzeniami. Sceny składają się z różnych obiektów, które będą zawierały pewne komponenty, które nadają obiektom różne cechy lub zachowania.

Rozwój funkcjonalności jest szerokim zadaniem, ale może być podzielony na kilka procesów: Projektowanie i Architektura, Modelowanie Danych, Integracja SDK i Zarządzanie Zdarzeniami.

### 5.1. Projekt i architektura

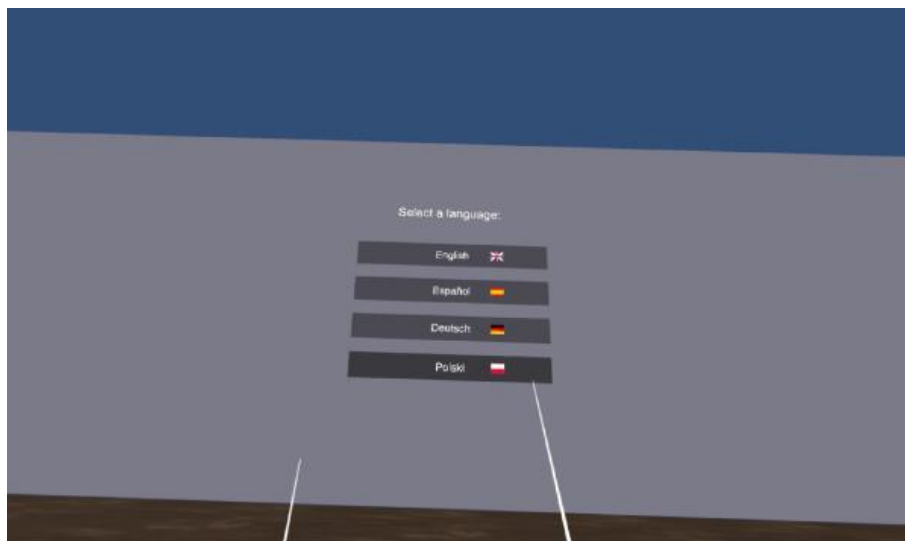
Jest to proces kształtowania scenariusza opracowanego w poprzednich częściach, czyli obmyślenia ścieżki i podejmowania decyzji, które pociągnie za sobą każda scena lub ekran.

Doświadczenie narzędzia składa się z dwóch odrębnych części. Struktura, którą przedstawi, jest pokazana na schemacie blokowym na rysunku 9.

#### 5.1.1. Główne menu

Pierwszy ekran pozwala użytkownikowi dostosować się do wirtualnego środowiska, w którym będzie uczestniczył podczas korzystania z narzędzia. Chociaż ruchy są ograniczone, aby skupić uwagę użytkownika na prawdziwym celu sceny, którym jest wybór jednej z sytuacji proponowanych dla prawidłowego użycia nano produktów. Konieczne jest również wybranie języka, w którym użytkownik chce, aby aplikacja została uruchomiona.

Kolejny ekran będzie zależał od wyboru dokonanego na tym ekranie.



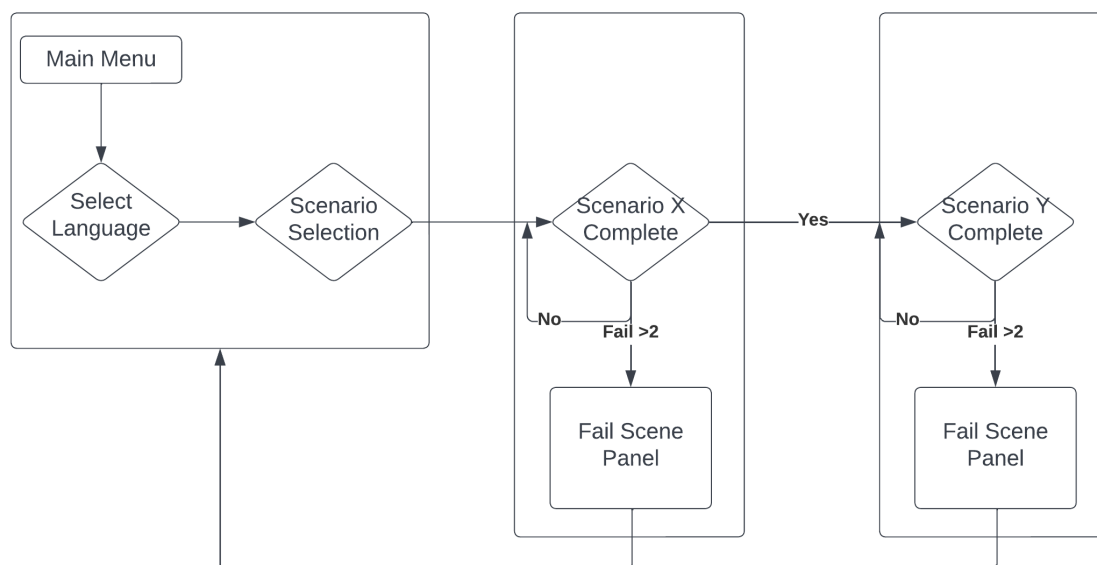
(a)



(b)

Rysunek 12: Główne menu (a) Wybór języka (b) Wybór sceny

Scena: Po wybraniu przez użytkownika konkretnego scenariusza, który chce przetestować, użytkownik zostanie przeniesiony do tej sceny z pływającym ekranem powitalnym, gdzie zostaną wyszczególnione szczegóły sceny i wymagania dotyczące zadania. Jeśli użytkownik nie wykona zadania, może je powtórzyć 3 razy z rzędu; kolejne niepowodzenia spowodują przekierowanie go do zdobycia lepszej wiedzy teoretycznej przed ponowną próbą. Jeśli użytkownik zda, może przejść do następnego scenariusza i tak dalej.



Rysunek 13: Diagram .

## 5.2. Modelowanie danych

W tym zadaniu celem jest wygenerowanie ogólnego schematu, na którym przechowywane są i pobierane informacje. W tym celu wykorzystano moduł PlayerPrefs, czyli klasę udostępnianą przez Unity, która umożliwia przechowywanie danych w postaci klucza z nazwą projektu.

Zmienne te w aplikacji Oculus przechowywane są jako plik XML wewnątrz katalogu /data/data/pkg-name/shared\_prefs/pkg-name.v2.playerprefs.xml, gdzie pkg-name będzie nazwą aplikacji.

Dodatkowo Unity zapewnia bardzo łatwą dostępność do tego zbioru danych dzięki funkcjom udostępnianym przez klasę PlayerPrefs.

Jedyną zmienną, która będzie wspólna i którą będziemy musieli przechowywać, ze względu na strukturę tej aplikacji, będzie ta, która odnosi się do języka. W tym celu wykorzystamy następujące funkcje:

- **SetString** (nazwa, wartość): Funkcja ta pozwoli nam na zapisanie wartości łańcuchowej na zmiennej name. W tym przypadku zostanie wpisany referencyjny kod ISO języka wybranego w menu.
- **GetString** (nazwa): Funkcja ta zwraca wartość zmiennej nazwa. Będzie ona wykorzystywana w każdej ze scen, aby wiedzieć, który język został wybrany.

## 5.3. Integracja SDK

Stworzenie dowolnego typu aplikacji Virtual Reality wymaga trzech elementów. Kompatybilnego urządzenia, które można podłączyć do komputera, silnika graficznego do gier wideo oraz zestawu SDK (Software Development Kit). Pierwsze dwa wymagania zostały wyjaśnione na początku rozdziału i będą to odpowiednio Oculus i Unity.

SDK to dowolny zestaw narzędzi, które umożliwiają twórcom oprogramowania w procesie tworzenia. W przypadku Wirtualnej Rzeczywistości, a w szczególności pakietów adaptowalnych do Unity, jest ich kilka, w tym jeden specyficzny dla Oculus. Wybór padł jednak na XR Interaction Toolkit. Pakiet opracowany przez Unity, który pozwala na dostosowanie projektu lub aplikacji do każdego rodzaju urządzenia. Innymi słowy, nawet jeśli narzędzie jest opracowane z Oculus Quest, może być zainstalowane na dowolnym urządzeniu kompatybilnym z Unity. Otrzymujesz więc szerszy zasięg.

Aby zbudować projekt, który wprowadza SDK i umożliwia interakcję w Unity, należy wykonać następujące kroki:

1. Otwierając silnik wygenerować projekt z szablonem Universal Render Pipeline. Ważne jest, aby zrobić to z tym szablonem, ponieważ zapewnia on zoptymalizowaną grafikę, a jest to wymóg Wirtualnej Rzeczywistości, jeśli chcemy mieć poprawne doświadczenia.
2. Zainstaluj SDK, czyli pakiet XR Interaction Toolkit. Można to zrobić z okna Package Manager, dostępnego z opcji menu Window.
3. Będziemy musieli wskazać typ urządzenia, na którym będzie budowany. W tym celu należy włączyć opcję VR-supported w Project Settings/Player.

W ten sposób projekt Unity jest gotowy do tworzenia Virtual Reality. Aby jednak rozpocząć interakcję urządzenia ze środowiskiem graficznym, należy wykonać kroki opisane w dalszej części rozdziału.

## 5.4. Zarządzanie zdarzeniami

Zarządzanie zdarzeniami można uznać za najbardziej rozbudowane zadanie w ramach rozwoju funkcjonalności. Polega ono na wykorzystaniu komponentów dostarczonych przez Unity i wygenerowaniu niewielkich skryptów w celu osiągnięcia oczekiwanej interakcji użytkownik-maszyna.

Ten rozdział mógłby być zbyt długi, ponieważ każda scena ma wiele specyficznych szczegółów na poziomie interakcji. Ale tutaj zostaną wyszczególnione tylko wspólne i ważne punkty, które pozwolą osiągnąć cel.

### 5.4.1. Przygotowanie scen do VR

Po przygotowaniu projektu Unity i skonfigurowaniu lub graficznym przygotowaniu scen, kolejnym krokiem będzie przygotowanie sceny tak, aby urządzenie interakcyjne mogło być połączone z kamerą, a my mogliśmy użyć jej do testów w trakcie rozwoju. Proces wyjaśniony w poniższych krokach zostanie powtórzony dla każdej ze scen, które zostaną wykorzystane w projekcie:

1. Pierwszą rzeczą jest usunięcie domyślnej kamery.
2. Wygeneruj pusty obiekt z komponentem XR Rig, który będzie kontenerem dla wszystkich interakcji człowiek-maszyna. Nazywamy go VR-Rig.
3. Wygeneruj obiekt dziecięcy, również pusty, który będzie pozycją referencyjną, w której użytkownik rozpocznie interakcję. Obiekt ten będzie się nazywał Camera Offset.

4. Wygeneruj kamerę jako dziecko Camera Offset, z komponentem Tracked Pose Driver. Jest to obiekt, który będzie pełnił rolę naszych oczu oraz dodany komponent, który pozwoli nam się obracać.
5. Po tym wypełnij zmienne komponentu XR Rig (obiekt macierzysty, VR-Rig) pozycją referencyjną i utworzoną kamerą. Oraz umieść floor jako wartość atrybutu Tracking Origin Mode, sprawi to, że kamera będzie się automatycznie dostosowywać do naszej wysokości. W ten sposób otrzymujemy w pełni funkcjonalne oczy systemu.
6. Aby uzyskać mobilność i widoczność kontrolerów, generujemy dwa nowe dzieci wewnątrz obiektu Camera Offset i dodajemy atrybut XR Controller.
7. Dodajemy model, który chcemy wykorzystać jako kontrolery w atrybucie Model Prefab.
8. Na koniec do tych dwóch obiektów dodajemy komponent XR Direct Interactor, który nada kontrolerom interaktywność z resztą obiektów w scenie.

#### 5.4.2. Kontrolery

Jest to proces, w którym uzyskuje się wartości generowane przez naciśnięcie każdego z przycisków na pokrętlach. Będzie to wykorzystywane do ogromnej liczby zadań, takich jak wybieranie obiektów, zaznaczanie opcji itp.

Przed wygenerowaniem czegokolwiek z tymi wartościami, ważne jest, aby wiedzieć, jaki typ danych obsługuje każde wejście. Można to zobaczyć w oknie XR Interaction Debugger, do którego można się dostać z Window/Analysis.

Mając te informacje, kolejnym krokiem jest wygenerowanie skryptu, który pozwoli uzyskać te wartości. Klasa, która została wygenerowana dzięki temu nosi nazwę HandController i jest używana jako komponent każdego z modeli 3D, które zostały dodane w atrybucie Model Prefab kontrolerek. Najważniejsze punkty kodu to:

```
public class HandController : MonoBehaviour
{
    //Input
    private InputDevice targetDevice;

    //Controller model options
    public bool showController = false;
    public List<GameObject> controllers;
    private GameObject spawnedController;

    //Device characteristics
    public InputDeviceCharacteristics controllerChar;

    //Hand model
    public GameObject handModel;
    private GameObject spawnedHandModel;

    private Animator handAnimator;

    //Get if is Right or left
    public string isRight;

    //Input variables
    public float trigVal;
    public bool primaryButton;
    public float gripValue;
    public Vector2 axisVal;
```

Rysunek 14: HandController class



1. Pierwszą rzeczą, którą widzisz na powyższym obrazku, jest deklaracja zmiennych. Te, które są publiczne, będą odpowiadały zmiennym atrybutom komponentu.
2. Podczas generowania nowego skryptu zawsze automatycznie tworzone są dwie funkcje: Start(), która jest wykonywana raz na początku instancjonowania, oraz Update(), która jest wykonywana iteracyjnie przez cały czas życia komponentu.
3. Pierwsze co dzieje się wewnątrz tego komponentu to funkcja TryInitialize(), w niej otrzymywane jest urządzenie VR według jego charakterystyki. Dla każdego kontrolera cechą będzie to czy jest on prawy czy lewy. Uzyskuje się to poprzez wywołanie funkcji GetDevicesWithCharacteristics() wewnątrz modułu InputDevices.

```
//Try get input device and attach the selected model to it
void TryInitialize()
{
    //Get VR devices by characteristics
    List<InputDevice> devices = new List<InputDevice>();
    InputDevices.GetDevicesWithCharacteristics(controllerChar, devices);

    if (devices.Count > 0)
    {
        //Get target device and get a controller model
        targetDevice = devices[0];
        GameObject prefab = controllers.Find(con => con.name == targetDevice.name);

        if (prefab)
        {
            spawnedController = Instantiate(prefab, transform);
        }
        else
        {
            Debug.LogError("Did not find corresponding controller model");
            spawnedController = Instantiate(controllers[0], transform);
        }

        //Instantiate Hand Model and get Animator component
        spawnedHandModel = Instantiate(handModel, transform);
        handAnimator = spawnedHandModel.GetComponent<Animator>();
    }
}
```

Rysunek 15: Funkcja TryInitialize()

4. Następnie wykonywana jest funkcja Update(), która wywołuje funkcje UpdateEvents() i UpdateAnimations().
5. UpdateEvents() odpowiada za uzyskanie każdej z wartości wygenerowanych przez kontroler poprzez funkcję TryGetFeatureValue() i zapisanie ich w zmiennych publicznych, które zostały wygenerowane w pierwszej instancji.
6. UpdateAnimations() jest funkcją stworzoną w celu aktywowania animacji, które każdy z przycisków zakładałby w przypadku posiadania rigowanego modelu ręki. W każdym razie nie jest to funkcja istotna w zadaniu uzyskania wartości.



```
//Update hand events
void UpdateEvents()
{
    //Trigger event
    if(targetDevice.TryGetFeatureValue(CommonUsages.trigger, out float triggerVal))
    {
        trigVal = triggerVal;
    }

    //PrimaryButton event
    if (targetDevice.TryGetFeatureValue(CommonUsages.primaryButton, out bool primary))
    {
        primaryButton = primary;
    }

    //Grip event
    if (targetDevice.TryGetFeatureValue(CommonUsages.grip, out float gripVal))
    {
        gripValue = gripVal;
    }

    //Joystick event
    if(targetDevice.TryGetFeatureValue(CommonUsages.primary2DAxis, out Vector2 axis))
    {
        axisVal = axis;
    }
}
```

Rysunek 16: Funkcja UpdateEvents().

#### 5.4.3. Manager Quiz'u

Aby ułatwić i przyspieszyć rozwój, wygenerowana została klasa o nazwie QuizManager, która ma za zadanie ujednolicić proces tworzenia paneli. Element ten składa się z następujących kroków:

1. Generowane są zmienne dla wszystkich pytań, aktualnego pytania, poprawnej odpowiedzi oraz elementów UI.
2. Generowana jest funkcja InitQuiz(), która odpowiada za inicjalizację każdej ze zmiennych, która będzie potrzebna. Następnie wywołą funkcję SetNextQuestion(), aby zainicjalizować quiz.
3. SetNextQuestion() i SetAnswerOptions(): Funkcje te aktualizują zmienne i elementy panelu, takie jak teksty i przyciski, ustawiając wartość następnego pytania w kolejce, usuwając poprzednie, jeśli takie było
4. Funkcja, która jest uruchamiana po wciśnięciu przycisku odpowiedzi to SelectButton(Button but). W przypadku pytania z tylko jedną odpowiedzią wywoła ona funkcję CheckResponse().
5. Checkresponse (): Jest to funkcja z logiką, która odpowiada za sprawdzenie, czy wybrana odpowiedź jest prawidłowa. Jeśli jest, to przejdzie do następnego panelu, jeśli nie, to doda awarię do sceny i da początek albo panelowi awarii, albo panelowi awarii sceny.



Rysunek 17: Widok Quiz'u

#### 5.4.4. Tłumacz

Proces ten działa na każdej ze scen w tle, aby dostarczyć każdemu elementowi UI narzędzia teksty przetłumaczone na wybrany język.

Najpierw przeprowadzono tłumaczenie tekstów na każdy z języków partnerów uczestniczących w projekcie. Dokumenty te będą miały format, w którym każde słowo lub fraza będzie posiadała klucz, który pozwoli nam na dostęp do tej frazy w każdym z języków.

W tym celu zostały utworzone dwie klasy:

- *MainTranslate*: Klasa ta nadzoruje tworzenie połączenia pomiędzy skryptami tłumaczącymi a środowiskiem pracy, w tym przypadku kodem. Za każdym razem, gdy zostanie wykonane wyszukiwanie tego klucza wygenerowanego w poprzednim skrypcie, klasa ta zwróci słowo lub frazę w odpowiednim języku.
- *SceneTranslate*: Klasa ta nadzoruje wskazywanie na początku każdej nowej sytuacji na MainTranslate, który jest językiem wyszukiwania wybranym w menu.

Mając te procesy na miejscu, wszystko, co musisz zrobić, to zmienić każdą z linii tekstu, którą zamierzałeś wprowadzić do następnej linii kodu:

```
text = MainTranslate.Fields[textKey];
```

Rysunek 18: Okno tekstowe

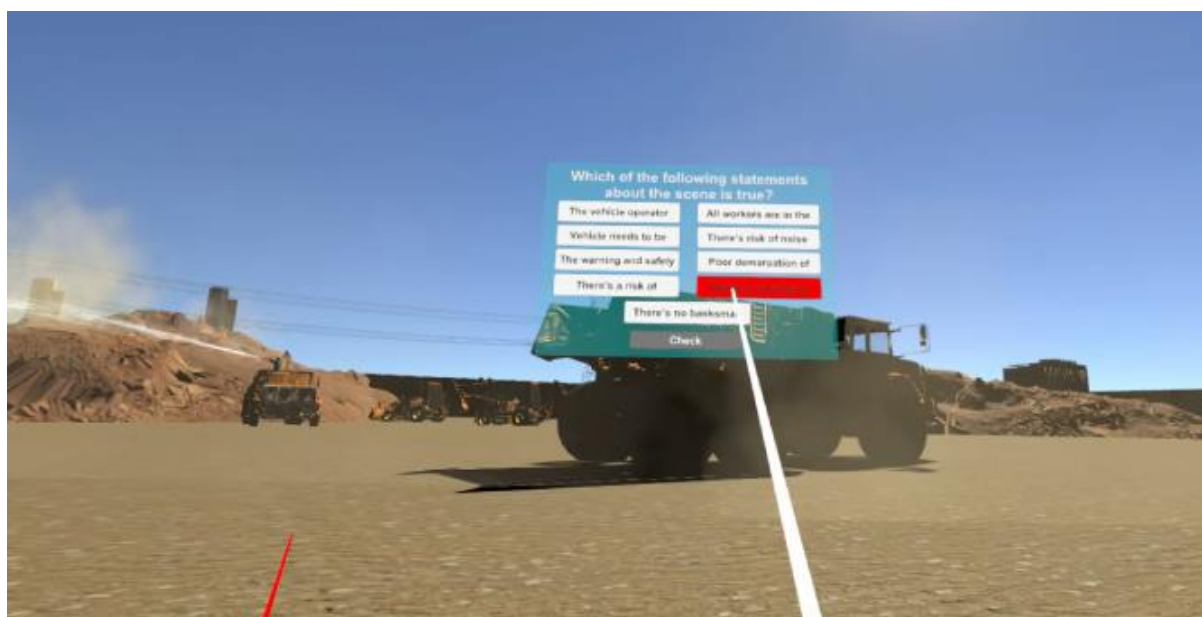
#### 5.4.5. Interakcja

Zadanie to pozwala użytkownikowi na interakcję z obiektami lub panelami znajdującymi się w pewnej odległości. Polega ono na wygenerowaniu wiązki, która wychodzi z dłoni i pozwala użytkownikowi wykonywać działania na określonych obiektach. Dla tego projektu bardzo ważne będzie udzielenie odpowiedzi na postawione pytania.

Aby wygenerować tego typu interakcję, konieczne będzie wykonanie następujących kroków:

1. Utwórz obiekt Ray Interactor, który pojawi się w menu GameObject/XR dla każdej z rąk.
2. Następnie wybierz obiekty, na które ma oddziaływać. W tym celu zmień parametr Raycast Mask komponentu XR Ray Interactor. Atrybut ten da nam możliwość wyboru warstwy, na której chcemy, aby promień działał, dzięki czemu obiekty, na które chcemy, aby promień działał, będą musiały mieć tę warstwę.
3. W tym przypadku wybraliśmy warstwę UI paneli dostarczanych przez Unity.
4. Na koniec, aby promień nie pojawiał się cały czas, będziemy musieli zmienić atrybut Invalid Color Gradient komponentu XR Interactor Line Visual na wartość transparentną. Tak, aby tylko wtedy, gdy ręka wskaże na panel, pojawiła się błyskawica.
5. Te dwa obiekty zostaną wprowadzone jako Camera Offset, obiektu utworzonego w ramach przygotowania sceny, tak aby znalazł się on w obiekcie odwotującym się do urządzenia VR.

Dzięki implementacji tego punktu otrzymujemy funkcjonalność pozwalającą na generowanie każdego z różnych opisanych paneli i wchodzenie z nimi w interakcję.



Rysunek 19: Obraz interakcji

## 6. Podsumowanie

Głównym celem projektu jest stworzenie bardzo innowacyjnego, interaktywnego środowiska szkoleniowego opartego na technologii wirtualnej rzeczywistości (VR), aby przekazać pracownikom budowlanym niezbędne umiejętności i edukację w zakresie interakcji z maszynami i materiałami, które zostały osiągnięte poprzez rozwój narzędzia SafeCRobot przedstawionego w niniejszym raporcie. Niniejszy raport przedstawia metodologię wykorzystaną do opracowania narzędzia SafeCrobot, które jest immersyjnym wirtualnym narzędziem szkoleniowym w zakresie bezpieczeństwa. Świadomość zagrożeń i potrzeb bezpieczeństwa związanych z automatyzacją stopniowo staje się niezbędną umiejętnością w obecnych i przyszłych miejscach pracy w wyniku złożonych interakcji narzuconych przez powiązane maszyny i materiały w tym samym środowisku pracy. Narzędzie to pozwoli różnym interesariuszom z sektora budowlanego (stowarzyszenia zawodowe, związki zawodowe, administracja) na osiągnięcie pożądanej świadomości i podniesienie kwalifikacji w zakresie wpływu robotyki i automatyki na zdrowie, bezpieczeństwo i środowisko w branży budowlanej w całej Europie.