SEMESTER SHOWDOWN:

THE ACADEMIC

ADVENTURES

The learning platform based on
gamification for the assimilation
of knowledge related to Courses 7-9

Onea Maria-Teodora

Roÿ u Alexandru-Ionuÿ

Your father was Theodor

Mitria Alexandru

Group 322AC

FACULTY OF COMPUTER SCIENCE

Polytechnic University of Bucharest

content

1. Project objectives

The main objective of this project is to design a gamification application, through which users can test their knowledge of Project Management.

Thus, this platform represents a contribution to the digitization of education, offering modern and accessible ways to the new generation of students.

This application is part of the BOOSTING SUSTAINABLE DIGITAL EDUCATION FOR EUROPEAN UNIVERSITIES project, in collaboration with the Erasmus+ program.

1.1 Why Escape Room?

The idea of escape rooms has become very popular lately, both virtual and physical. This game format trains the participant's critical thinking, as it requires increased attention to details, making logical connections between clues and, above all, the ability to react quickly in borderline situations. Thus, we consider that
implementing an application in this format can be beneficial to the deepening of knowledge that is more difficult to assimilate.

1.2. Team tasks

Given the complexity of this project and the disadvantageous number of members in our team, we decided to handle most of the tasks together.

- Onea Maria-Teodora: Documentation, Progress Monitoring, Map Design, Question Scripts + Question Scene Design
- Roÿu Alexandru-Ionuÿ: Map scripts, Creating the Menu + Options (related settings), Special script for opening doors based on an access code, map design

- Tÿnase Theodor: Map Scripts, Map Design, Pause Menu, Question Scripts, Special script for opening doors based on an access code, Map design
- Mitria Alexandru: Character + Mouse Controls, Documentation, Map Design

2. Analysis / Documentation

2.1. The advantages of gamification-based learning platforms

Educational games are not a new concept, as they have existed for a long time, being used especially by publishing houses that create mini-games for the little ones (Editura Edu, Doxi, etc.). The effect they have on the quality of children's deepening of knowledge denotes their increased efficiency, as they help to make logical connections between different concepts in a fun way.

Thus, it is not at all surprising that schools and universities are approaching the same tactic to attract the attention of students. Platforms such as Kahoot!, EdApp or Gametize offer such services, contributing to the efficiency of teaching in educational institutions.

A first advantage of these platforms is the creation of a relaxing and animated atmosphere that increases the attention of the participants. The introduction of interactive elements intensifies active learning at the expense of passive learning.

Another advantage is the possibility to incorporate teamwork into the learning process. Some platforms can be used by a larger group of people in a team format. By combining the game with teamwork, students retain in a much faster manner certain concepts that can be complicated to assimilate in the classic way.

Although this approach presents significant advantages, it cannot completely replace classical teaching, the two combined being the ideal option for an effective educational process.

2.2 Escape Room: The Basics

Escape Room is a game where one person/group of people discovers clues, solves puzzles and completes missions in one or more rooms within a limited time. The goal is to get out of the room he's in, but the game isn't just about escaping, it's also about uncovering certain secrets and making connections between pieces of information.

These physical games take their inspiration from escape room video games, taking the concept and of course the basic rules. These rules differ from one game to another, depending on the specific purpose and conduct of the game. However, the condition to be met in order to win is to solve all the puzzles in the allotted time.

Basically, players have to explore to find important objects or clues, and then piece together the information obtained. In some cases (Escape Room

Horror), players are forced to find innovative ways to free themselves from captivity or hide from "dangerous" people.

If the participants get stuck, they can ask for hints from the Gamemaster, receiving advice that can help them clarify certain tasks.

The most common tasks found in an escape room are puzzles of various forms: physical (correct positioning of certain objects in a system), coded messages based on an alphabet, riddles, etc.

3. Justification of the solution

In our project, we decided to keep certain elements characteristic of escape room games, such as: the need to explore several areas of the room (in our case, the building), locked doors, interactive objects, etc.

Also, the game is made in First-Person style, thus being an experience immersive.

From a design perspective, I chose a darker theme to match the format of the app. In this sense, the setting and appearance of the building in which the game takes place is a damaged and unkempt one, similar to a stage used in physical escape room games.

The basic principle of our app is inspired by other games of this genre: there are several rooms that the user has to enter and look for clues to get the codes of the locked doors. In our case, we replaced the clues with sets of questions that must be filled in correctly to move on.

Having said that, the changes we made were made with the aim of integrating the learning process within such a game.

3.1. Technologies used

For this project we decided to use Unity Engine over other graphics engines, as it is very intuitive and easy to use. Unity Engine is a graphics engine that can be used to create 2D or 3D games, as well as interactive simulators. It is based on scripts in C#, also using "drag and drop" functionality.

Also, Unity is a cross-platform engine, being supported on Windows, macOS, Linux, but also on mobile platforms (iOS, Android, etc.).

Since we used this software, we used the tutorials on its official website to create the base game, adding components later

authentic. In addition, we discovered a multitude of tutorials on different platforms such as Youtube, which helped us develop the app according to our ideas.

Another advantage besides the intuitive use of the graphics engine was the existence of an official virtual store of useful and easy-to-integrate assets. Thus, we were able to customize the game to our liking without having to create certain objects or textures ourselves.

4. Implementation description

4.1 The principle of the game

The solution proposed by our team includes the division of the project into several main components that must be implemented: map, character, menu, options and establishing the questions that appear in the game.

Being a minimalist program, the controls we interact with within it are strictly limited to character movement and pausing the game. Thus, we can move on the map with the WASD keys (W-forward, A-left, S- back, D- right) SHIFT – to run, SPACE – to jump the objects present in the game, and to put when the game is , with E we interact with paused, press the Esc key (in the pause window we have access to the menu, we can also exit the game directly).

- W: face;
- A: left;
- S: back;
- D: right;
- SHIFT: run;
- SPACE: jump;
- P: game break;
- E: interactivity with objects;

As a gameplay, it spans three rooms connected by a hallway. In each room there are 3 questions that the player must answer in order to move on.

The first room will be unlocked, and after completing the set of questions in it, the player will receive the access code for the next room.

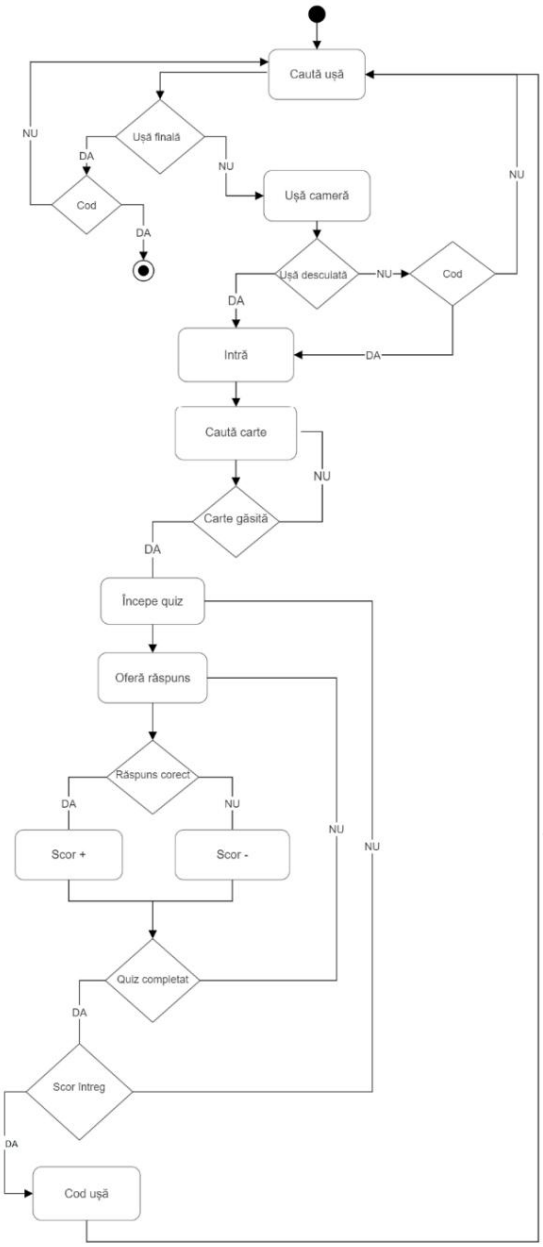
This code will be entered in a window that will automatically appear when the player interacts with that door. After entering the code, he will press the E key to confirm. The door will open and the next set of questions can begin.

There is a room where there are 2 sets of questions, each of them giving a snippet of the passcode for the last room.

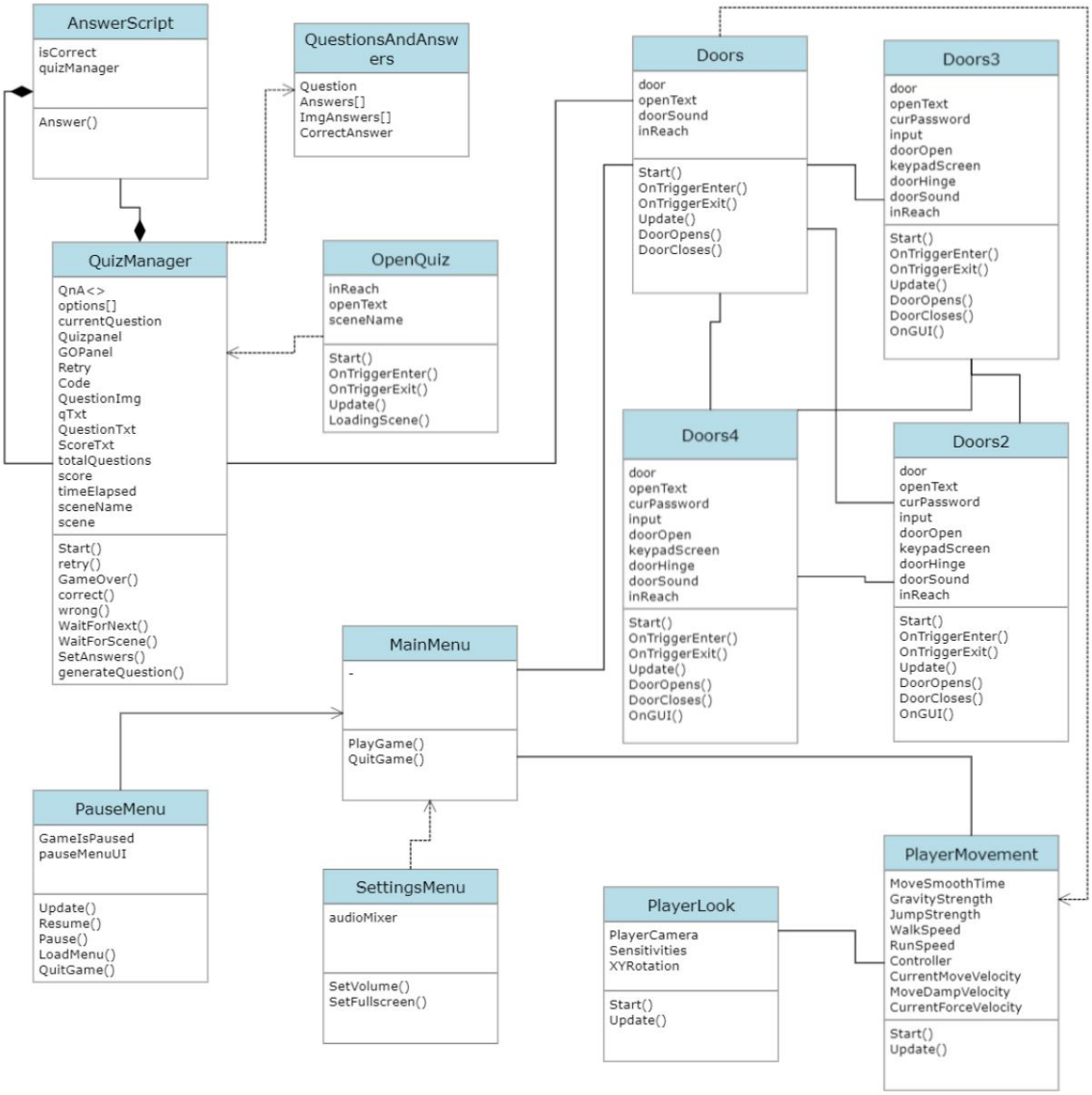
At the end, the user will receive the access code for the exit door from the building, and after opening it, the game will be over.

4.2. UML diagrams

Activity chart



Class diagram



4.2. OOP elements used in the project

Within the project, we find the following notions of Object Oriented Programming (POO):

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine; using
UnityEngine.Audio;

public class SettingsMenu : MonoBehaviour {

        public AudioMixer audioMixer;
```

```
    public void SetVolume (float volume) {

        audioMixer.SetFloat("volume", volume);
    }

    public void SetFullscreen (bool isFullscreen) {

        Screen.fullScreen = isFullscreen;
    }
}
```

Class: The SettingsMenu class is an example class in OOP. This class represents a settings menu in a game and contains methods and attributes for managing settings.

Attributes (or fields): Within the SettingsMenu class, we have a single attribute, audioMixer, of type AudioMixer. This is used to store a reference to an audio mixer, which can be used to adjust the volume of sounds.

Methods: The SettingsMenu class contains two methods, SetVolume and SetFullscreen. Methods are functions defined inside the class and are used to perform specific actions.

Using objects: The SetVolume method takes a float (volume) argument and uses the audioMixer object to set the volume of the sounds based on the provided value.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class Doors : MonoBehaviour {

    public Animator door;
    public GameObject openText;
    public AudioSource doorSound;
    public bool inReach;

    void Start() {

        inReach = false;
    }

    void OnTriggerEnter(Collider other) {

        if (other.gameObject.tag == "Reach") {

            inReach = true;
```

```
                openText.SetActive(true);
            }
        }

        void OnTriggerExit(Collider other) {

            if (other.gameObject.tag == "Reach") {

                inReach = false;
                openText.SetActive(false);
            }
        }

        void Update() {

            if (!(door.GetBool("Open")) && inReach &&
Input.GetButtonDown("Interact")) {

                DoorOpens();
            }

            else if (door.GetBool("Open") && inReach &&
Input.GetButtonDown("Interact")) {

                DoorCloses();
            }
        }

        void DoorOpens() {

            Debug.Log("It Opens");
            door.SetBool("Open", true);
            door.SetBool("Closed", false);
            doorSound.Play();

        }

        void DoorCloses() {

            Debug.Log("It Closes");
            door.SetBool("Open", false);
            door.SetBool("Closed", true);
        }
}
```

Encapsulation: Class attributes are declared public so they can be accessed from outside the class. In OOP, it would be better practice to declare attributes as

private or protected and use accessor methods (getter and setter) to manipulate them. However, in this case, the attributes are public so they can be set from the Unity editor.

Inheritance: This principle was mainly used within question scripts to retrieve our chosen questions and answer choices for use in calculating the final score and updating the quiz window.

Composition: often used within the project to link a class to objects of other classes, for example: in PlayerMovement.cs we refer to the CharacterController class (which is already created in Unity) through an object called Controller.

Events and Actions: The OnTriggerEnter(Collider other) and OnTriggerExit(Collider other) methods are used to respond to collision events with other objects in the game.

Flow control: In the Update() method, an if control structure is used to check whether the door is open or closed and whether the player is within interaction range. This determines what action will happen when the player presses the interaction button.

GUI and Visual Actions: The Animator class is used to control the animation of doors (opening and closing), which adds visual elements to the game.

4.3. Game features

Through the Unity graphics engine, we were able to implement several essential functionalities for any game, these are:

- Menu: the starting window of the game, through which we can access the following scenes:
  - ÿ Play, which starts the game;
  - ÿ Options, where we can change the sound volume in the game and we can check or uncheck the fullscreen option;
  - ÿ Help, where the purpose and rules of the game are explained, as well as the controls; also there are two useful tips to make the player experience easier
  - ÿ Quit, which closes the application
- Pause window: the user can pause the game by pressing the P key, at which point a pop-up opens with the following buttons:
  - ÿ Resume: pressing this button resumes the game from where it left off
  - ÿ Menu
  - ÿ Quit

- Pop-up: to enter the access codes of the doors, when the user interacts with the door, a pop-up will appear in which the respective code can be written;

- Score: at the end of a quiz, the user can see how many correct answers he gave out of the total number of questions;

4.4. Testing

Testing is a crucial stage in the development of any application, as with its help we can discover possible bugs or interruptions that cause obstacles in using the program. Therefore, this process must be rigorous in order to eliminate as many inconsistencies as possible. In the case of our project, testing focused on the

accuracy of question scripts and interaction with important objects.

4.5 Performance indicators

To determine whether our application meets the required standards, we considered several performance indicators that helped us measure the performance of the project and identify issues that needed to be fixed. Thus, we selected the following performance indicators:

4.5.1 Memory used

The memory used by the app is in a decent range, being a small game though. Thus, it can be run on several different configurations without problems.

4.5.2 Storage space

As I mentioned before, our platform is small in size, as we only incorporated elements necessary for this type of application. Thus, the game does not require much storage space.

4.5.3 Number r of bugs

Following the testing process, we managed to eliminate a large part of the bugs encountered within our application. There are, however, some functionalities that are not fully resolved, but do not negatively affect the use of the platform.

4.5.4 Success rate

The percentage of players who manage to complete the game successfully will be tracked. This indicator can highlight the level of difficulty and effectiveness of the game in teaching Project Management content.

5. Conclusions

Despite the difficulties encountered, we believe that the final form of this project represents a solid foundation for a gamification application, and we have implemented a set of questions pertinent to the purpose of this project.

We believe that this project developed both our teamwork and logical thinking.
We were able to start from scratch and create rooms with details, textures and objects, which gave a realistic touch to the app.

This theme also forced us to be creative to find ways to implement the questions in an interesting way.