Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department

Game-Based Learning Model

July 2021

Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department

Game-Based Learning Model

By:
Mohamed Hatem Mohamed [CS]
Mohamed Ahmed Ahmed [CS]
Yousef Ahmed Mohamed [CS]
Mohamed Tarek Roshdy [CS]

Under Supervision of:

Dr. Sally Saad
Lecturer,
Computer Science Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

TA. Marwah Helaly
Assistant Lecturer,
Computer Science Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Abstract

The importance of the project lies in helping the students to understand the programming concepts in an interactive way with some entertainment to help them clearly know the basics of any concept which is not easy to understand through the traditional ways. Simulating the concept as a game story makes it easy to understand the complex concepts faster than learning them through the traditional ways. So, we put the learner in the right place to start learning and go deep in this field. After playing each level, we test his understanding of this concept by giving him a quiz about the level he already played. Starting from this point, we can simulate any field not only the programming fields, such as simulating dangerous fields like medical or pharmaceutical field. This will help the students to safely test any experiments or surgeries.

At this way, the learning process has become easier specially for the practical fields. Of Course, it does not cover everything about a chosen field, but it gives a clear simple overview about it and answers the basic questions of everything we learn: **Why we need this?** The answer is we need to learn this to solve a specific problem. As an example, the problem of recognizing the objects and differentiating between them by the software. How this could be done without image classifier. And **how we do this?** This question is answered along the whole game story, it will make the player do the solution steps on his own as simple as possible.

So, according to the implementation of our short vision, which is (level 1), we can say that the main features that are represented in the project are: Helping the player to understand the main pipeline/steps of solving the ML problems, putting these steps (data preprocessing, model training and testing) into an actual game story with continuous instructions for the player to know what to do at a specific time, giving a quiz to the learner at the end of the level to test his understanding, and providing the player with the external resources to help him going deep into this field and continue learning the theoretical part.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

CNN	Convolutional Neural Network
GBL	Game based learning
GUI	Graphical User interface
ML	Machine Learning
NY	New York
Q2l	Quest to learn
UI	User Interface

Introduction

Motivation

Interactive learning is a technique that seeks to make students actively engaged in the learning process, often using technology. This contrasts with more passive techniques like the traditional lecture. While the technological part of interactive learning can be intimidating to some, it is important to remember that technology exists to support pedagogy, not the other way around. With that in mind, instructors should evaluate educational technology with an eye toward tools that open exciting possibilities for their lessons and enhance learning for their students. Anything else should be left by the wayside.
And as **Sara De Freitas** - *Birkbeck College, University of London* - said in her paper **[2]:**

"*The next phase of focus upon educational games borrowed heavily from technology-enhanced learning approaches.*"

Interactive learning has several strategies, our project follows the strategy of a full-automated based educational software that will be represented as a game that simulates interactively the different technical concepts and algorithms. It will let the player clearly understand the implementation of a specific concept through a real-life related story that he will play. This process of learning will help the player to make a clear relationship between the real-life problems with their solutions and solving those problems technically. Gaming can create a dynamic process that can inspire learners to develop skills and competencies as they focus on the activities of the game.

In her article **[7], Bohyun Kim -** *Chief Technology Officer and Associate Professor at University of Rhode Island Libraries* **-** states that gamification is the process of game thinking and game mechanics to engage users and solve problems. The author states that games can help in learning because people are more motivated, engaged, and achieve more in games than in the real world and that games are designed to deliver an optimal experience to the user. Additionally, the author states that game dynamics can raise the level of user engagement with library resources and can be used to help users solve problems more effectively and quickly.

Problem Definition

Nowadays, we live in a world full of rapidly growing technologies and constant changes, and there are a lot of complex concepts that have appeared in the technical field and they are somehow complex to be understood through the traditional learning ways, so we need to renew the methods, usage, and the understanding of the concept of up-to-date education.

One of the modern educational curricula is **GBL**. It is a very outstanding and productive way of organizing the studying process, which is the curriculum of **Q2L** school in **NY** which is one of the most famous educational organizations that uses up-to-date approaches for arranging the educational process. That is why students in this school show significant results in their learning activities. So that this project aims to build game-based learning model that will work as a new method to understand some of the complex concepts.

Objectives

The following table shows the goals that we want to achieve through the project and the objectives to achieve these goals.

Table 1 : Objectives

GOAL	OBJECTIVE
Improve the traditional learning way	Build a new educational model that simplifies some of the complex concepts in the technical field.
Learning with entertainment	Implement an interactive game that applies the designed educational model using advanced machine learning techniques such as deep learning.
Simplifying the complex concepts	Simulate different technical concepts in a suitable and clear game story.
Achieving the desired learning outcomes	Design evaluation criteria that assure the full understanding of the specified subject.
Showing what to do then after knowing the basics of a specific subject.	Provide the player with external resources that will help him to continue his learning roadmap.

Time Plan

The following table describes the time plan of the project.

Table 2 : Time plan

Project Activities	Start Date	End Date
Survey	**1 Nov, 2020**	**30 Nov, 2020**
Requirement Specifications	**1 Dec, 2020**	**8 Dec, 2020**
Project Analysis Use Cases ER Diagrams Sequence Diagram	**9 Dec, 2020** **17 Dec, 2020** **25 Dec, 2020**	**16 Dec, 2020** **24 Dec, 2020** **31 Dec, 2020**
Project Design Project Architecture User Interface Design	**1 Jan, 2021** **16 Jan, 2021**	**15 Jan, 2021** **31 Jan, 2021**
Project Implementation	**1 Feb, 2021**	**31 May, 2021**
Project Testing Modules Testing Modules Integration	**1 Jun, 2021** **17 Jun, 2021**	**16 Jun, 2021** **30 Jun, 2021**
Project Documentation	**1 Nov, 2020**	**30 Jun, 2021**

Document Organization

Chapter two discusses the background of this project including a detailed description of the field of the interactive learning and specifically its application through the game-based learning with the scientific background related to it. It also reviews a survey of the work done in the field of game-based learning. Another part is reviewing the similar systems and discussing their description.

Chapter three explains the system analysis and the architecture design. It contains a description for the system modules, system users and users' characteristics.
The second part of this chapter contains the system design diagrams including a fully dressed use cases describing each function of the system, the class diagram with a description of all the main classes, the sequence diagram, and database schema.

Chapter four discusses the implementation and testing of the system. That includes a detailed description of all the functions in the system, a detailed description of all the techniques and algorithms implemented, a review of the UI design plan, storyboard, and wireframes, and finally the testing procedures and phases.

Chapter five describes in detail how to operate the project along with screenshots of the project representing all steps. It also includes an "Installation Guide" that would describe how to install the game, and all required third party tools that needs to be available for the game to run.

Chapter six includes the conclusion and the future plan for the game enhancement and the description of this projects' long vision.

Background

GBL is a type of game play that has defined learning outcomes. Generally, game-based learning is designed to balance subject matter with gameplay and the ability of the player to retain and apply said subject matter to the real world.

Game based learning describes an approach to teaching, where students explore relevant aspect of games in a learning context designed by teachers. Teachers and students collaborate in order to add depth and perspective to the experience of playing the game.

Good game-based learning applications can draw us into virtual environments that look and feel familiar and relevant. Within an effective game-based learning environment, we work toward a goal, choosing actions and experiencing the consequences of those actions along the way. We make mistakes in a risk-free setting, and through experimentation, we actively learn and practice the right way to do things. This keeps us highly engaged in practicing behaviors and thought processes that we can easily transfer from the simulated environment to real life.

The core concept behind game-based learning is teaching through repetition, failure, and the accomplishment of goals. Video games are built on this principle. The player starts off slow and gains in skill until they are able to skillfully navigate the most difficult levels. Games that are planned and designed well will offer enough difficulty to keep it challenging while still being easy enough for the player to win.

When a student works on game-based learning, they are probably just thinking that they're having fun with a game, but there's a lot that goes into crafting the game as a learning tool. The process begins with the curriculum and the core standards. Before the design even starts, a team of education, curriculum, and game experts decides on the exact lesson that will be taught to students.

"Learning games are built the same way a teacher builds a lesson plan. We start with a learning goal and then we work backwards." [9]
says **Suzi Wilczynski**, Founder and CEO of Dig It Games.

The theory of GBL says that not only the students! it involves a new way of training the employees of companies. We are talking about the use of games for learning. The offer for gamified content is increasing and getting more and more varied, with video games designed for nearly all target audiences and sectors.

At corporate level, this methodology is experiencing an undeniable boom. These are the principles upon which training through games is laid:

Constructivist learning

Game-based learning is built upon a constructivist type of learning. What does this mean? Constructivism posits the need to provide students with the necessary tools so they can build their own procedures in order to solve a problem. This implies a participatory process by students, who interact with their environment to solve the situation that is being set out to them.

Practice, experience, and interaction

Safe practice, experiential learning and interaction are the pillars upon which the theory of game-based learning stands. Learning through games allows students to experiment in non-threatening scenarios and acquire knowledge through practice and social interaction both with the environment and their peers.

Motivating approach

One of the strengths of game-based learning is its recognized capacity to capture the attention of students and ensure their full involvement: their engagement. The motivating approach of these games turns the learning process into something dynamic and interesting, whose appeal is maintained as students' progress to achieve objectives.

Encouragement of reflection

Besides motivation and a playful approach, GBL lays out situations that require reflection and decision making on the part of students in order to solve a problem. This way, the participant acquires knowledge and absorbs concepts while developing cognitive abilities derived from critical thinking, analysis of reality and conflict resolution.

Feedback and self-control

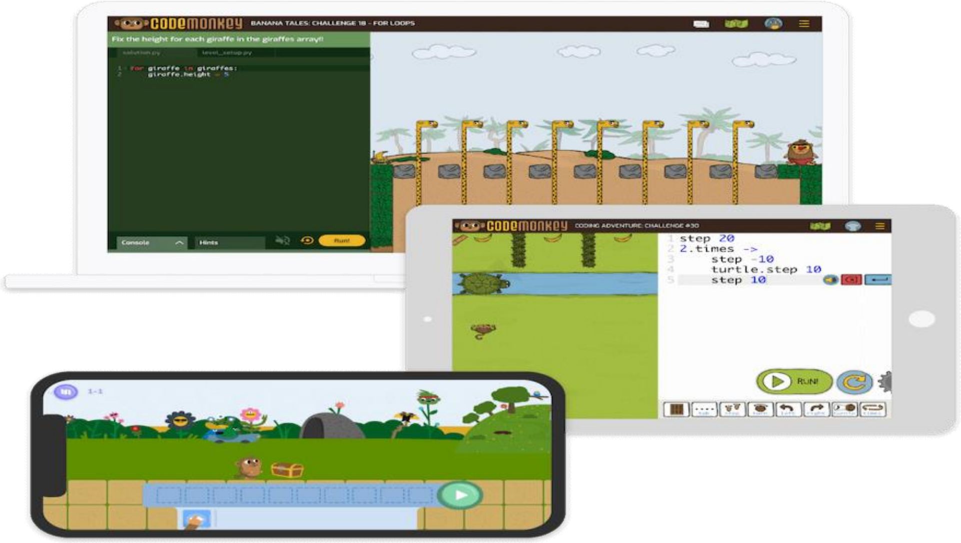
Unlike more traditional teaching methods, game-based learning allows the person being trained to have control of their own learning. Using serious games, students can get instant and personalized feedback about their knowledge, and everyone is aware of what they learn and what they must work harder at.

Creativity

The theory of game-based learning cannot ignore the benefit to creativity provided by the new ways of learning. The game encourages creativity and motivation by posing challenges and problems that students must solve using their imagination.

Here are some examples of existing similar systems:

1. **Code Monkey**


CODEMONKEY
BANANA TALES: CHALLENGE 18 - FOR LOOPS
Fix the height for each giraffe in the giraffes array!!
Console
Hints
Run!
CODEMONKEY
CODING ADVENTURE: CHALLENGE #30
RUN!
1-1

```
for giraffe in giraffes:
    giraffe.height = 5
```

```
step 20
2.times ->
    step -10
    turtle.step 10
    step 10
```

A fun and educational game-based web environment where kids learn to code without any prior experience. After completing CodeMonkey's award-winning coding courses, kids will be able to navigate through the programming world with a sense of confidence and accomplishment.

2. **OurCity**

A free Facebook city-building and civic education game. It is the product of a partnership between NetHope, the U.S. Agency for International Development (USAID), E-Line Media, Arizona State University's Center for Games and Impact, and local Jordanian companies and nongovernmental organizations.

The goal is to help young people develop the civic knowledge, awareness, and become engaged citizens to make communities stronger, healthier, and better able to meet the needs of the people who live in them. You can notice the visible inspiration drawn from the Sims.

The game allows the young players to build and develop the city from scratch with available resources and at the same time keeping all the townspeople happy.

3. Plantville

Siemens, the corporate giants implemented Plantville as an online marketing tool to showcase their products and services. The company uses the game as online recruitment tool and as part of employee training.

Plantville gives players the opportunity and challenge of running a complete virtual factory. Factory managers in Plantville are required to manage the hiring as well as deploying the workers with the tasks.

The players also get the additional responsibility of ensuring worker safety and satisfaction against production delivery schedules and to continuously adapt strategies to changing external conditions. The game also shows the ability to improve process efficiency with the purchase and installation of Siemens equipment.

Analysis and Design

System Overview

1. System Architecture

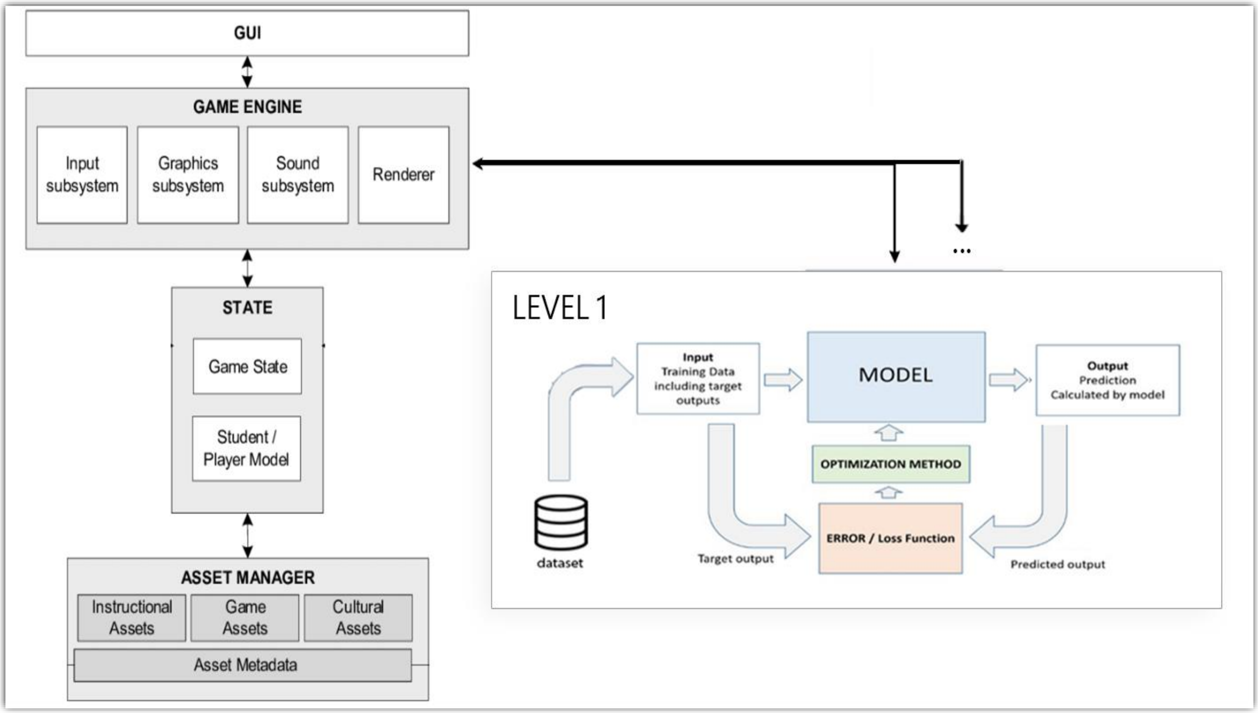


Figure 1 : System Architecture

As shown in Fig. 1 above, the overall functionality of the game was reduced into smaller, separate tasks which are handled by four major, self-contained components: a game engine, a state component, a Machine Learning model component and an asset manager. Many of these components provide the functionalities that Van Eck [12] states are needed for effective educational games.

1- The game engine is a mandatory component since it is in charge of the execution of the game. The presentation of audio and visual feedback, user input data captured from the GUI, scene updates, character positioning and so on are all controlled by the game engine.

2- Next, the state component records the circumstances of the game in the game state unit, and stores data about the student's knowledge, skill level, instructional activities, and general interaction with the game in the student/player model. Captured input data from the game engine is logged in the state unit.

3- The Machine Learning model that will take the inputs images from the dataset and start training the model passing throw Loss Function and optimization method until reaching the wanted accuracy.

4- Lastly, the asset manager handles the organization and distribution of all game-related, instructional, and cultural assets to the game engine via the game state. Appelman [13] comments that educational games should have more reusable content than entertainment games.

2. System Users

A. Intended Users:

1. *University students*
2. *School students with ages 12 – 18*

User Characteristics:

1. *Have a little technical background*
2. *Have beginner English level*

System Analysis & Design

Use Case Diagram

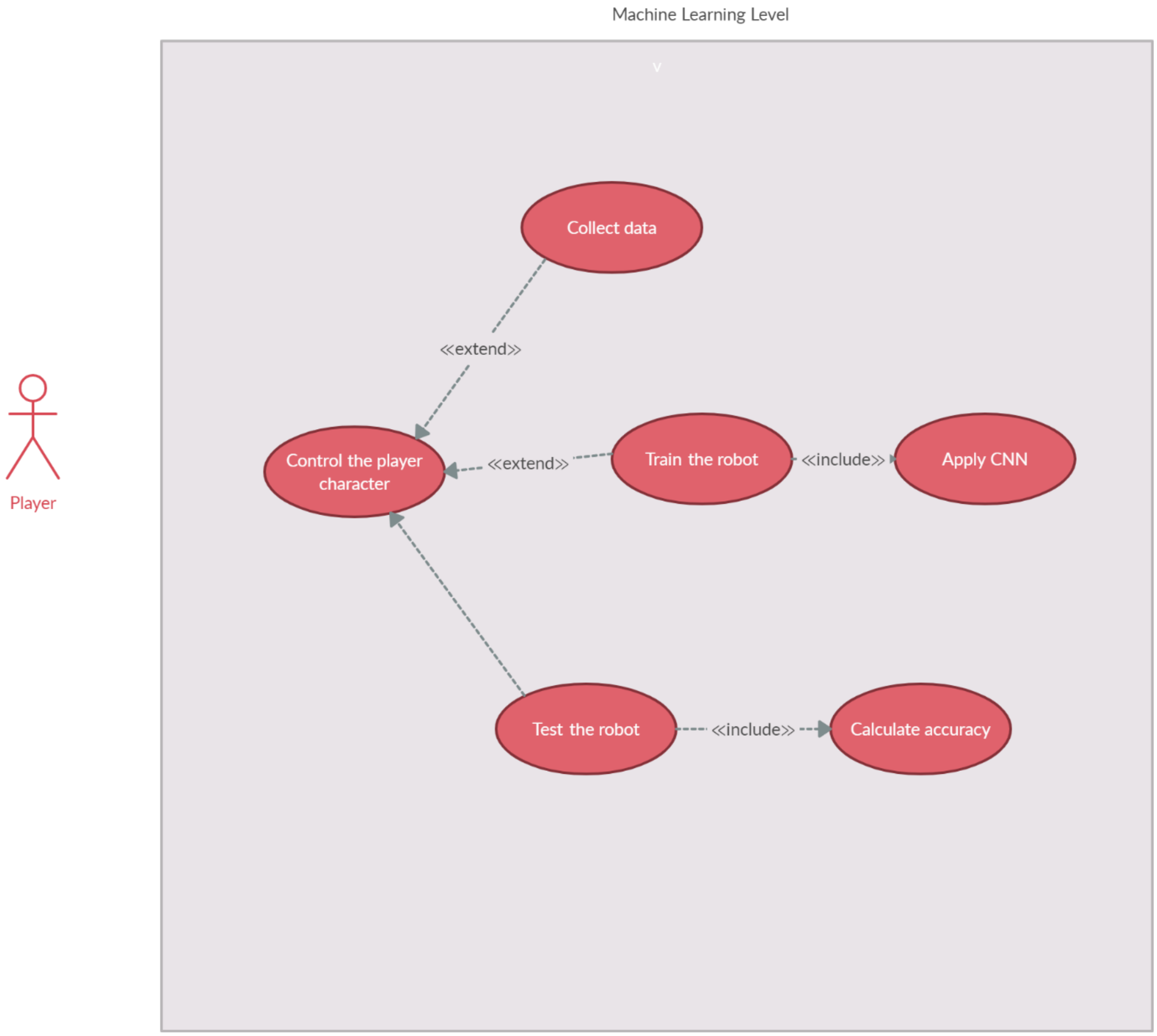


Figure 2 : Use Case Diagram

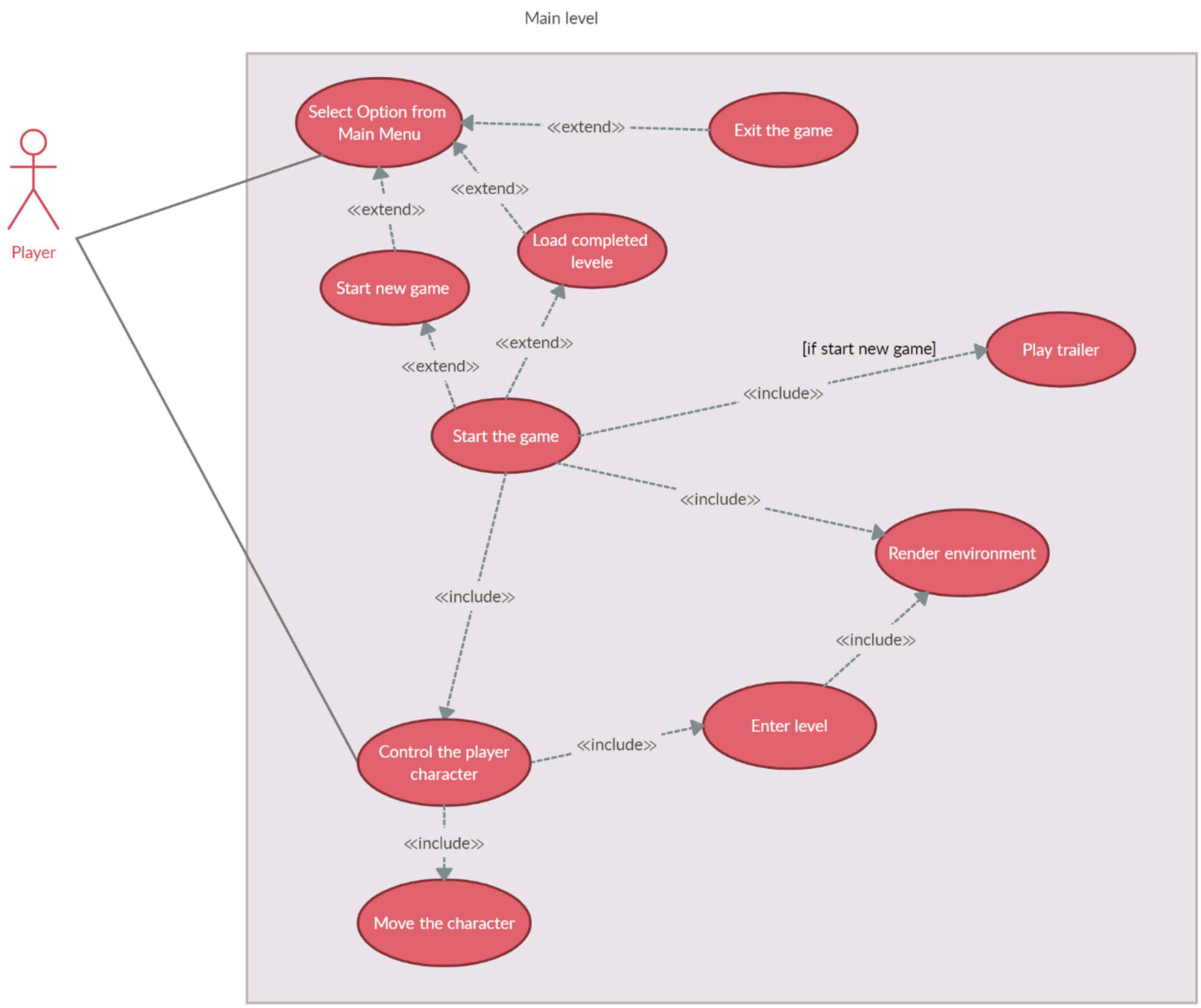


Figure 3 : Menu Use Case Diagram

Class Diagram

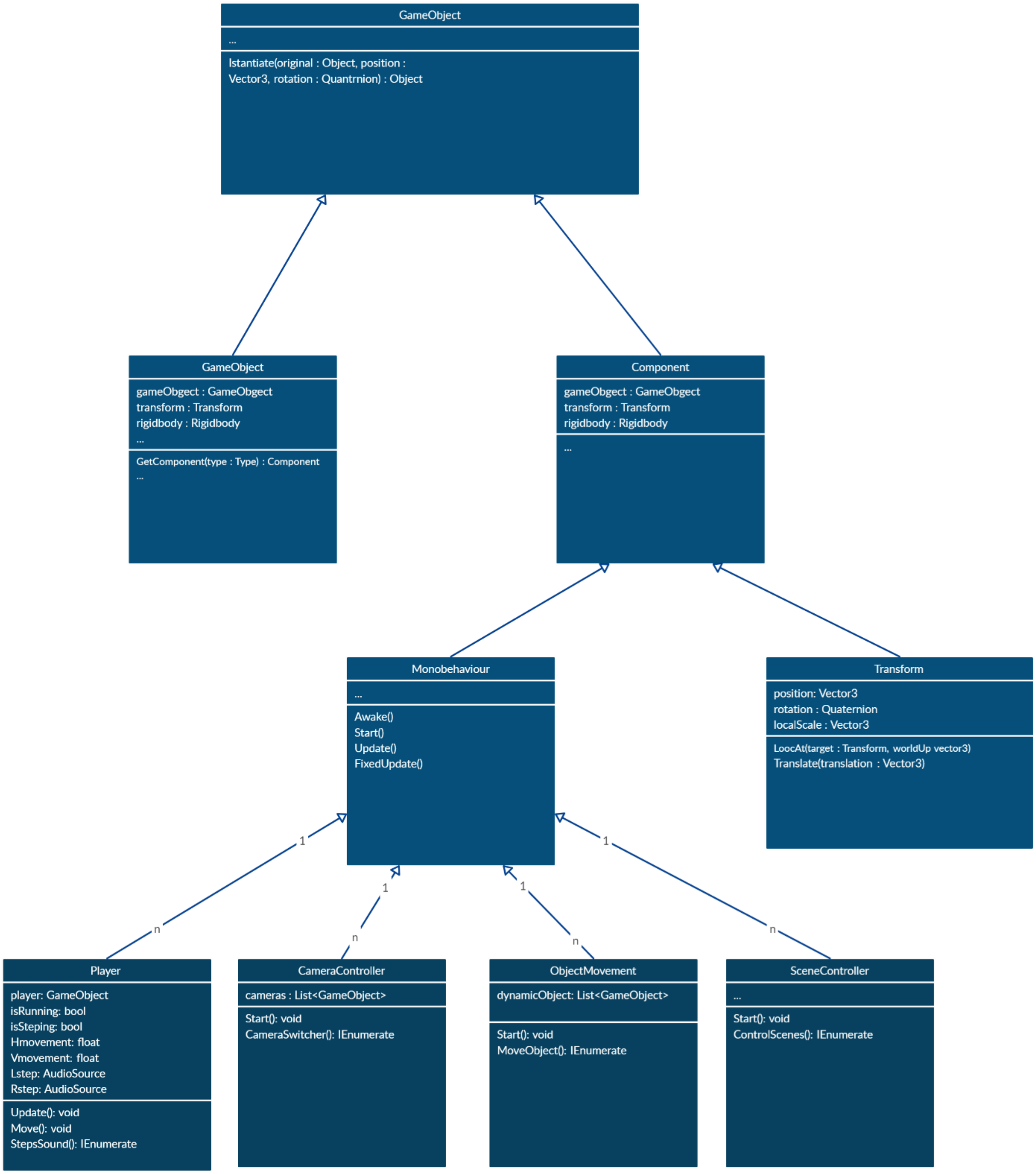


Figure 4 : Class Diagram

Sequence Diagram

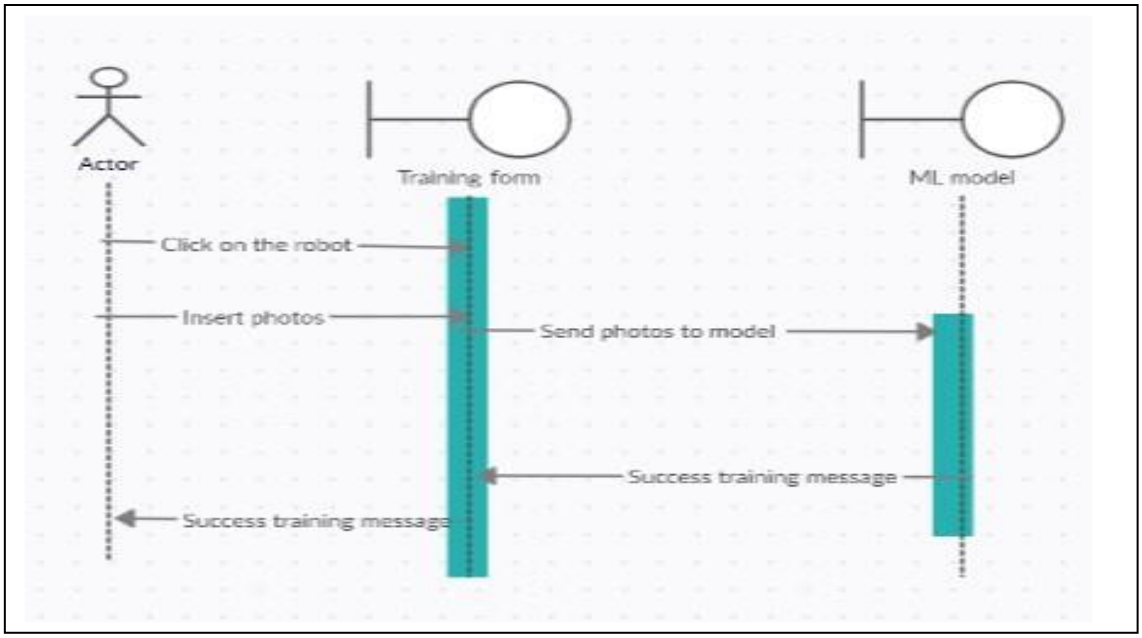


Figure 5 : Game Sequence Diagram

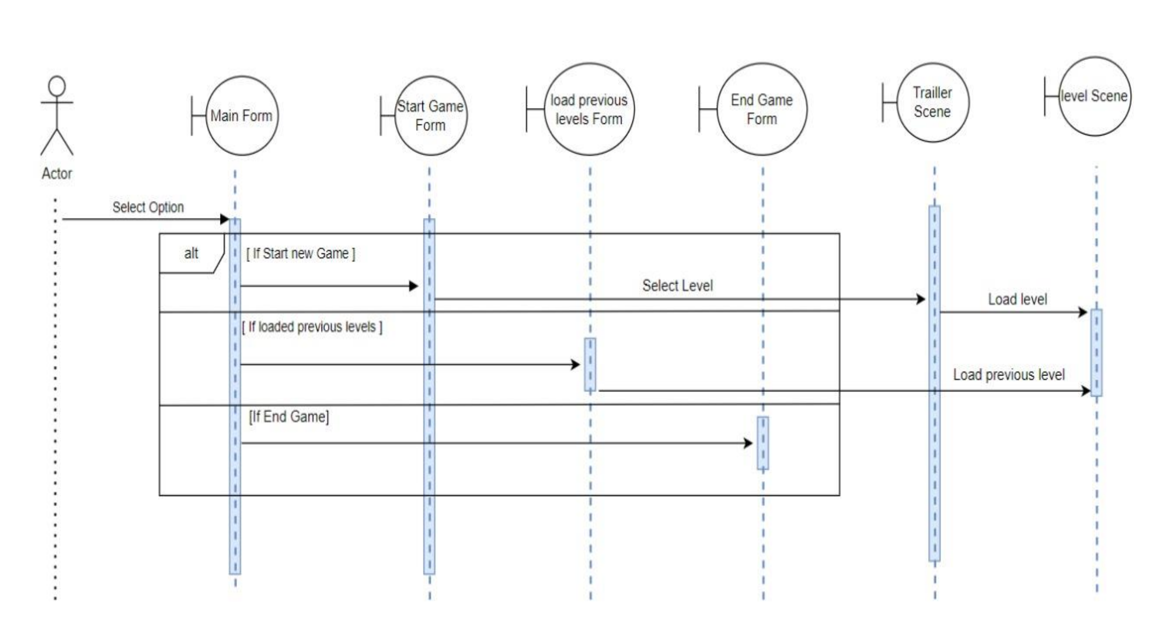


Figure 6 : Game flow Sequence

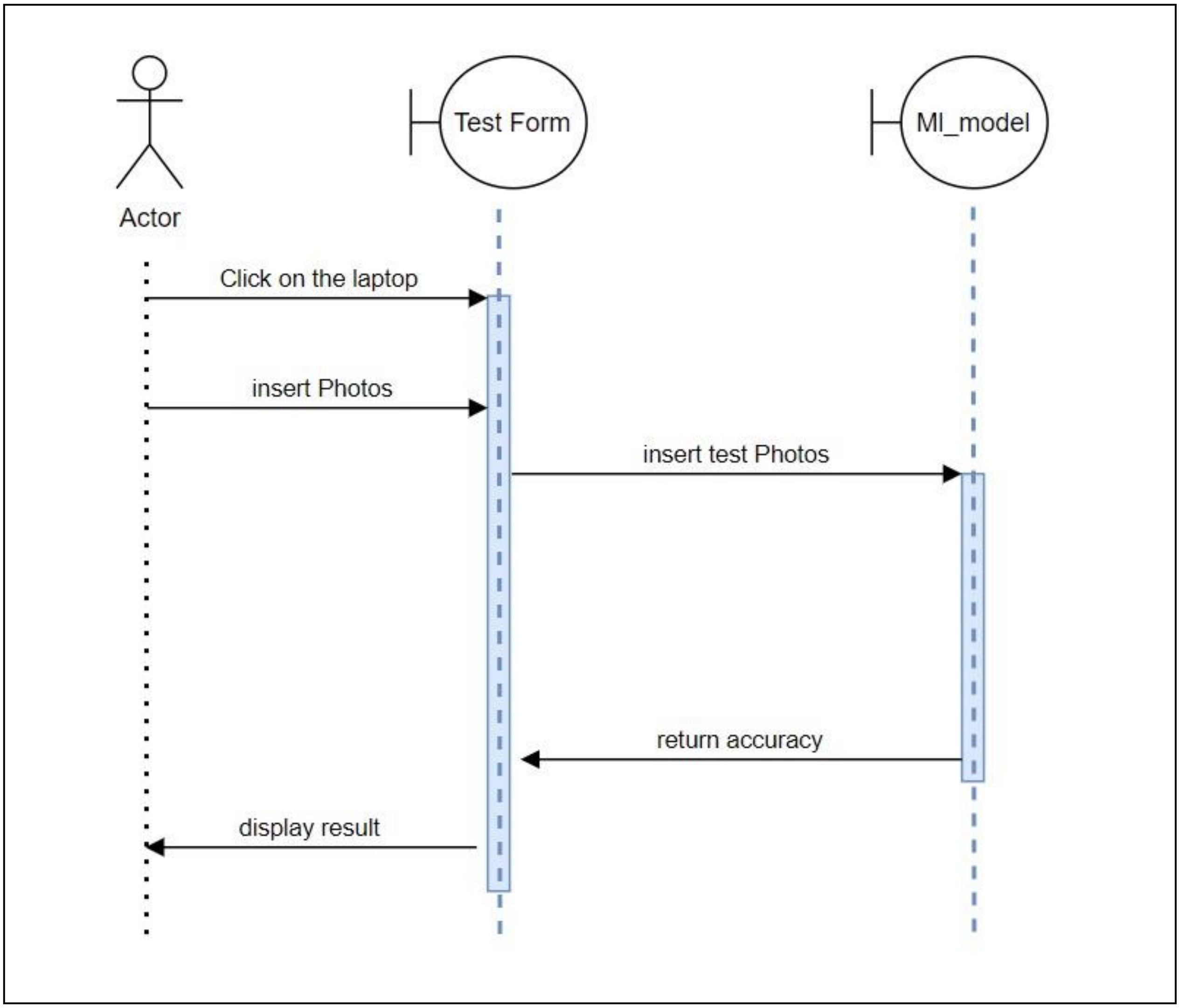


Figure 7 : Testing Sequence

Implementation and Testing

A detailed description of all the functions in the system:

- Scenes Navigation [Appendix 1]:
  - In Unity every Screen is a Scene, to switch from one scene to another this function needs to be implemented.
  - Give number for each scene in Unity then call a built-in function in unity with Scene number as a parameter to load it.

- Level Selection [Appendices 2, 3]:
  - ***Character Selection:*** user will choose from different characters, after choosing a character her/his description will appear.
  - ***Mission Selection:*** after character is selected by a selection button her/his missions will appear.

- Player Movement [Appendix 4]:
  - During Unity rendering Update () function is continuously called which updates the frames, so player movement is written in Update function and works as follow, when user presses one of movement buttons (W, S, A, D) the player will start to move.

- Player interactions [Appendix 5]:
  - To make player interacts with different objects like flash memory, cameras.... etc. player interactions are written in Update function and works as follow, user presses a button to apply interactions with objects then check collision with object, if a collision occurs interaction is applied.

- Camera Movement [Appendix 6]:
  - Attach camera with the player movement and synchronies the camera with mouse pointer.

- Data Split [Appendix 6]:
  - Data is a group of images in one folder, all images will appear to user in data split screen, then player splits them in the UI screen into three sets zombies training set, humans training set and testing set then press done button, each set is saved in a hash table.

- Data Organization [Appendix 7]:
  - To Train the model data should be organized as follow, for the two hash tables carrying the training sets two folders will be created for each class and for the hash table carrying the test set a folder will be created. All images in each hash table are copied to its created folder.

- Run Python script from C# [Appendix 8]:
  - Unity does not support python 3, so function is implemented to call python 3 from C# by giving it python3.exe path and python script path and return the outputs and errors of script as strings.

- Quiz evaluation:
  - After finishing the level, a quiz is provided to the user to solve and see the score to evaluate his understanding to the level he played.

- A detailed description of all the techniques and algorithms implemented:

  In the implementation of level 1, we have used a deep learning model which is CNN for the image classifier.

  - CNN Architecture [Appendix 9:16]:
    - First, Reading training images from the created folders (Humans and Zombies) in training list.
    - Second, vgg16 preprocessing is performed on data and images are resized.
    - Third, a Sequential model is built with the following structure:
      - Conv2d layers with 32 filters with size (3,3) and RELU activation function.

- Max pooling layer with filter size (2,2) and stride 2.
- Conv2d layers with 64 filters with size (3,3) and RELU activation function.
- Max pooling layer with filter size (2,2) and stride 2.
- Flatten layer.
- Fully Connected layer with activation function Softmax.

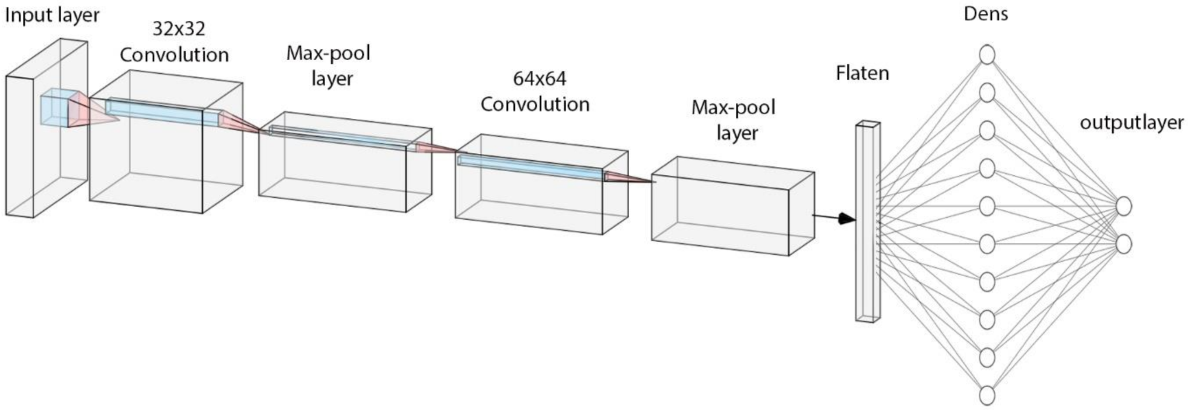


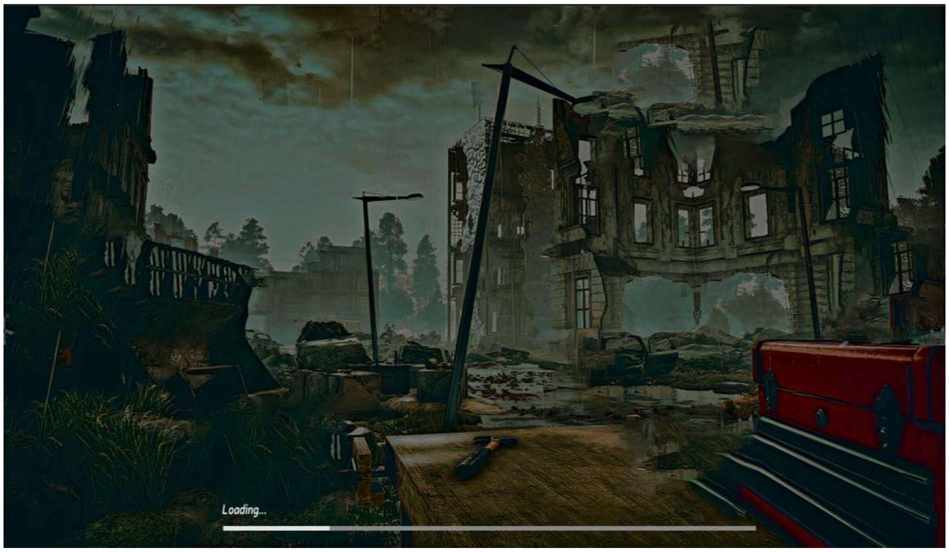
Figure 8 : CNN Architecture

- Model Train:
    - Model is trained by calling fit function which takes training list and steps per epoch and number of epochs as parameters.
    - Model is saved after training.

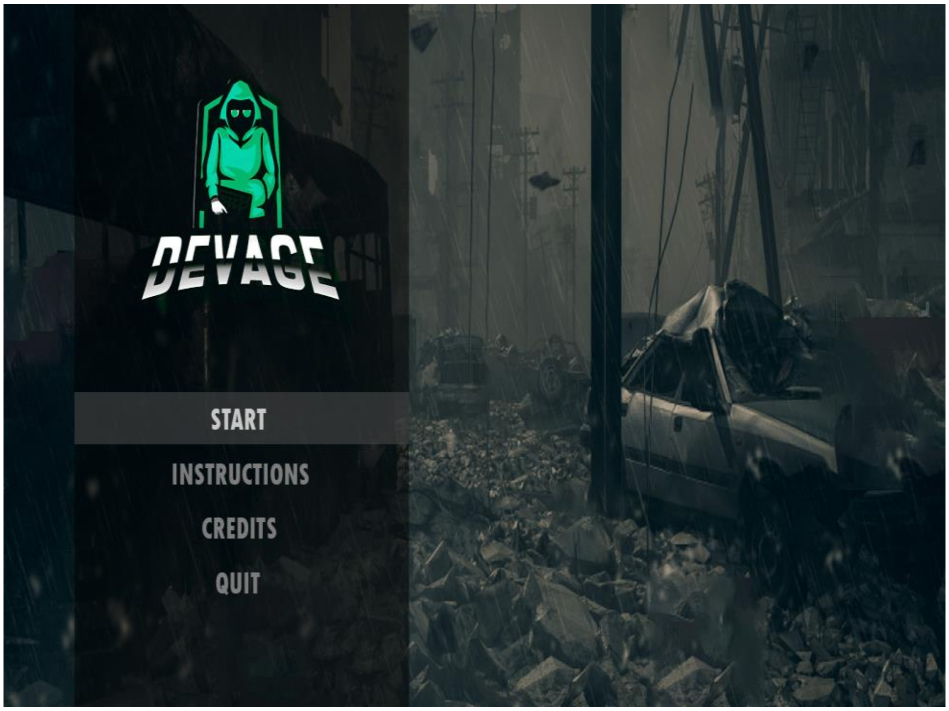
- Model Predict:
    - First, User select an image to predict from selected testing images, then the image is copied to a separate new created folder.
    - Second, Reading testing image from the created folder in testing list.
    - Third, the training model is loaded.
    - Finally, the Model start to predict by calling predict function that takes testing list and return the output (0 or 1).

- Description of any new technologies used in implementation.
  - Unity 3D integrated with python interpreter

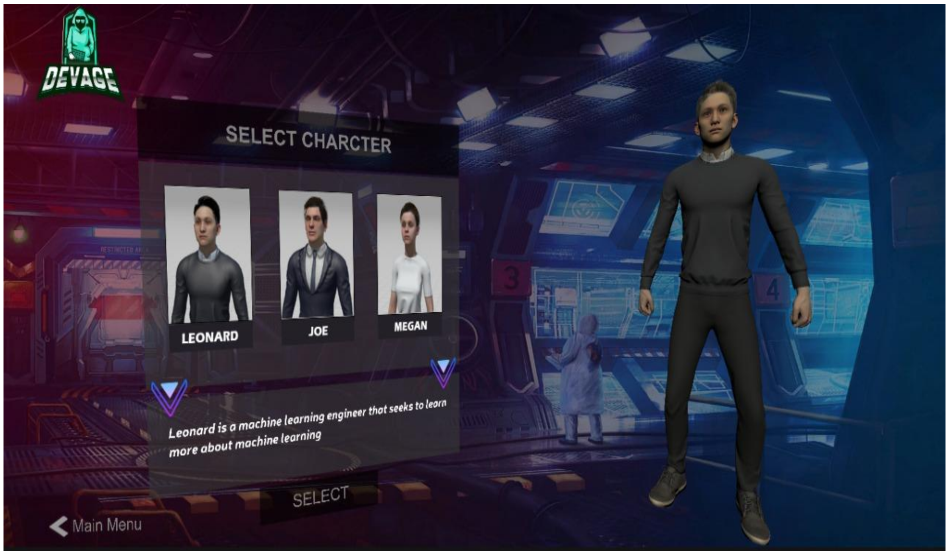
- UI Design and Wireframes
  - Loading Screen



  - Main Menu

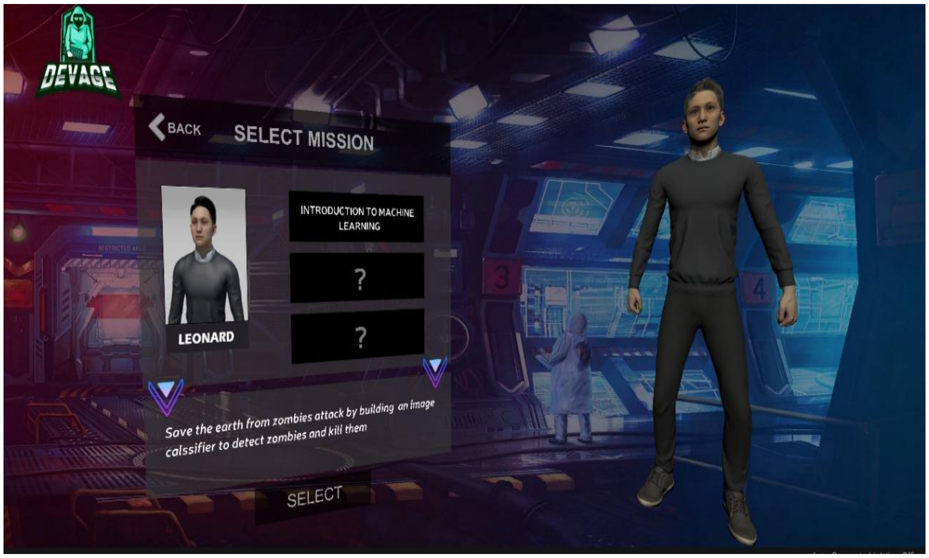


- Character Selection



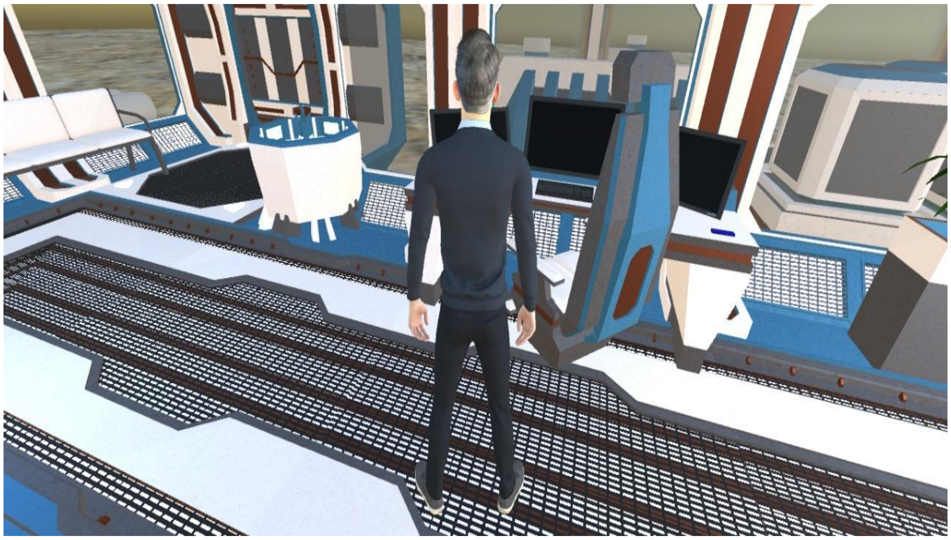
Leonard is a machine learning engineer that seeks to learn more about machine learning

- Mission Selection

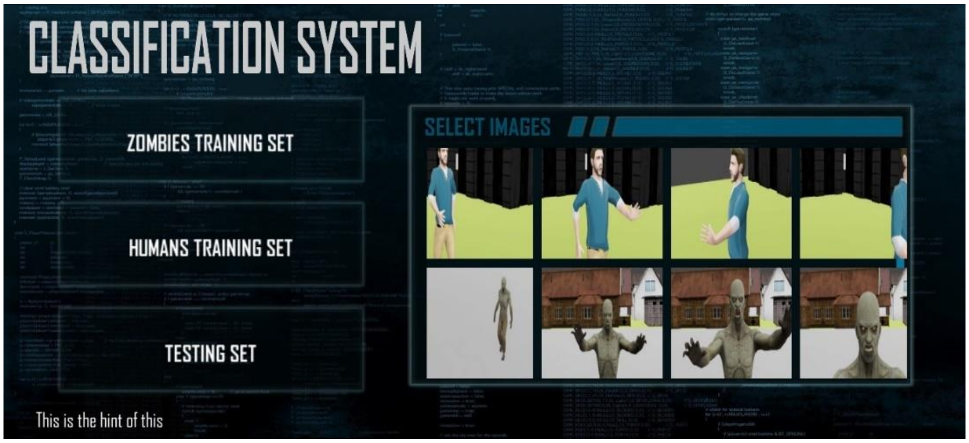


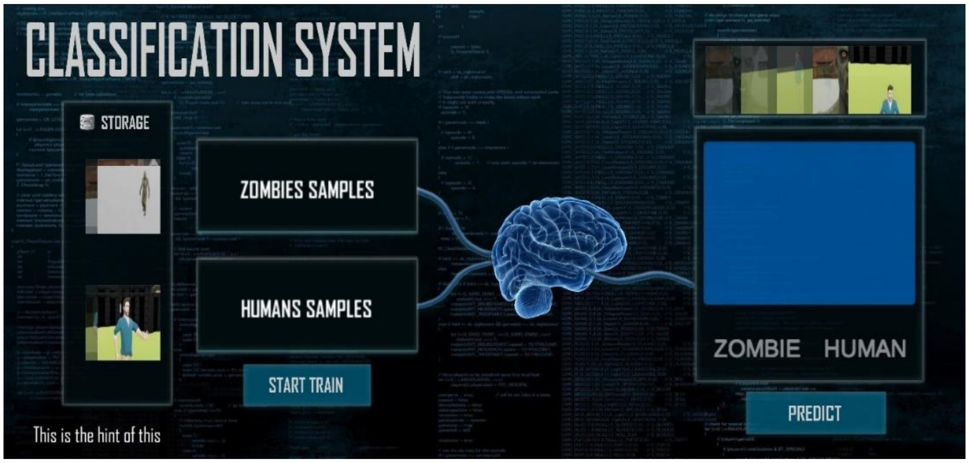
Save the earth from zombies attack by building an image calssifier to detect zombies and kill them

- Scene 1: Laboratory

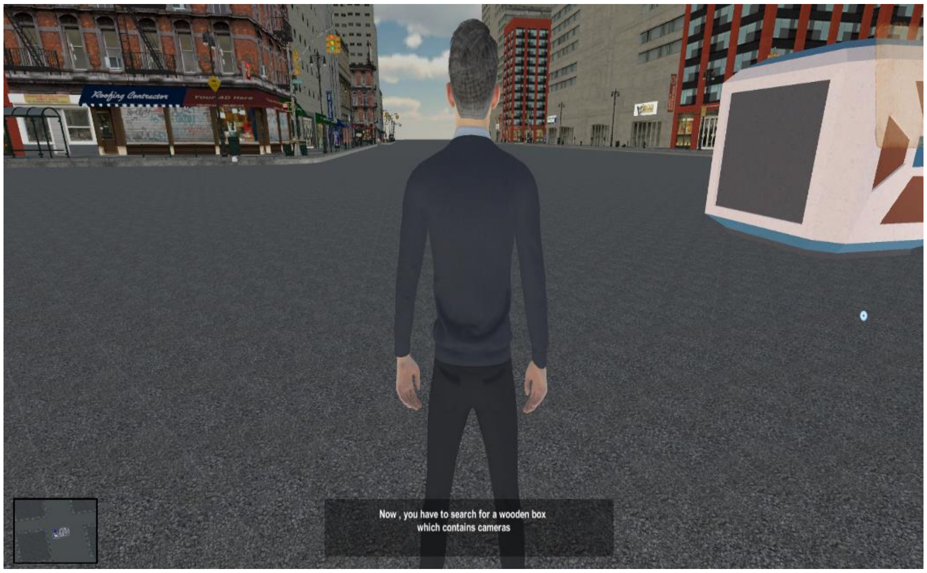


- Scene 2: Classification System





- Scene 3: Configure cameras in the city



Now , you have to search for a wooden box
which contains cameras

Testing procedures and levels used:

Unit testing:

- Test the model separately before using it in the game.
- Test each scene separately and check if the whole scene and the integration between scene’s components are working well or not.
- Test the character movement before using it.
- Test the user knowledge after completing each level.

Integration testing:

- Test the model with computer scene.
- Test the character movement with all scenes.

System testing:

- Test all the scenes together with all scripts.
- Test the user interactions with the game and check if it is completely made as it planned.

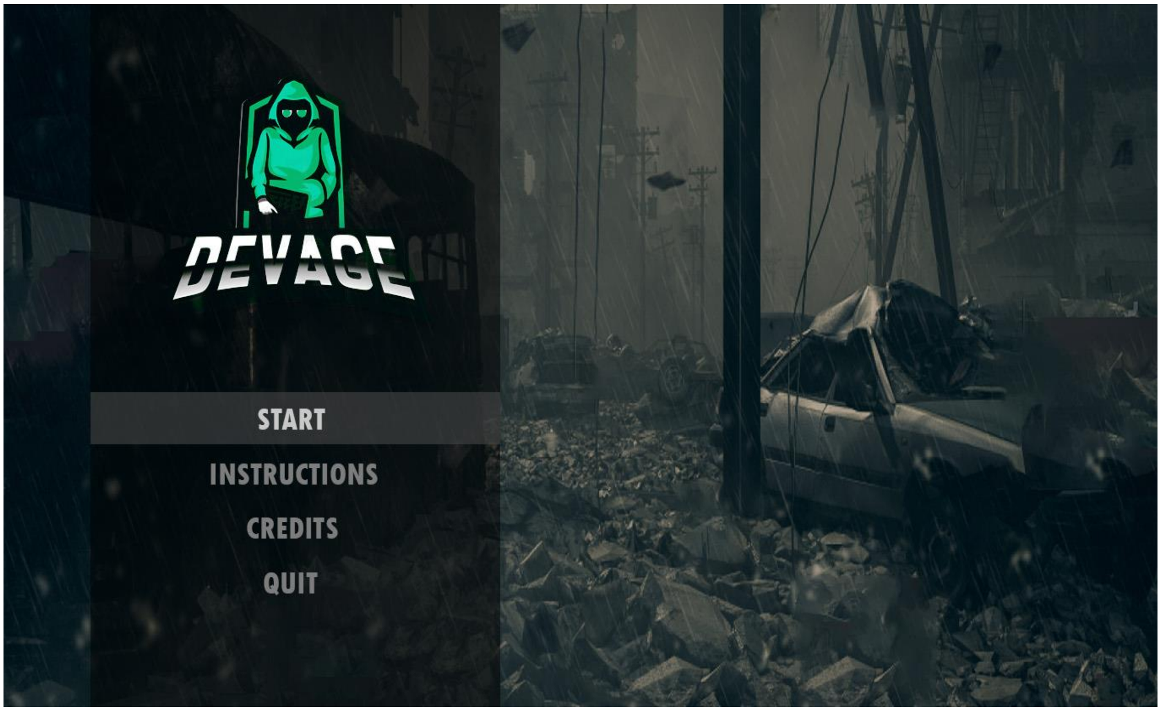
Acceptance testing:

- Give the learner a quiz after each completed level to check if he is completely understood the concept or not.
- Check the interactions between the learners and the game and check if there are any obstacles prevent them from continue learning the concept or not.
- Check if all the game is ready to be published or not.

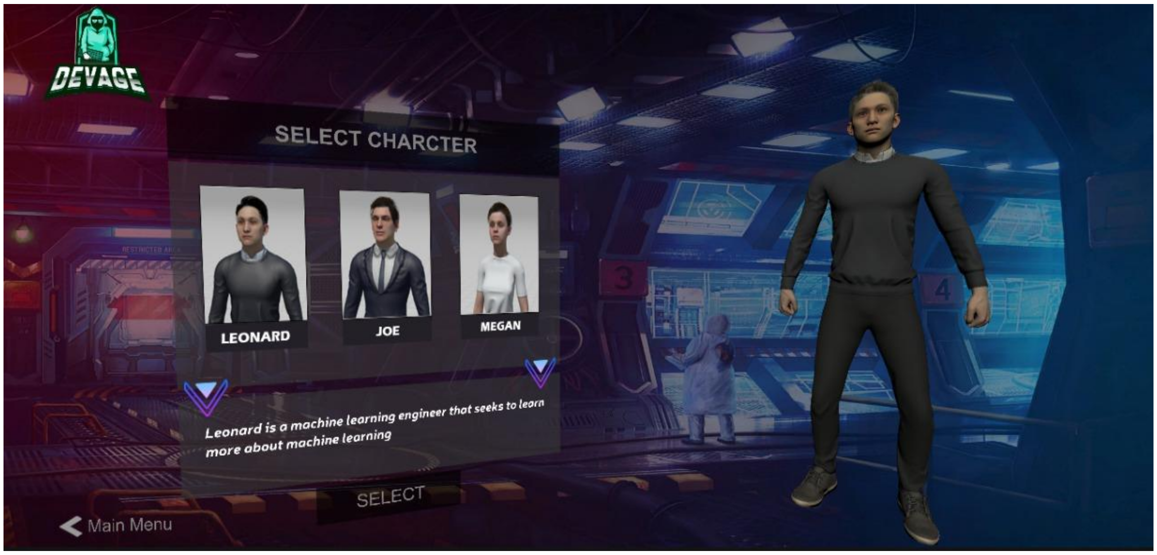
User Manual

1. Project Steps:

1. Select ‘Start’ from main menu

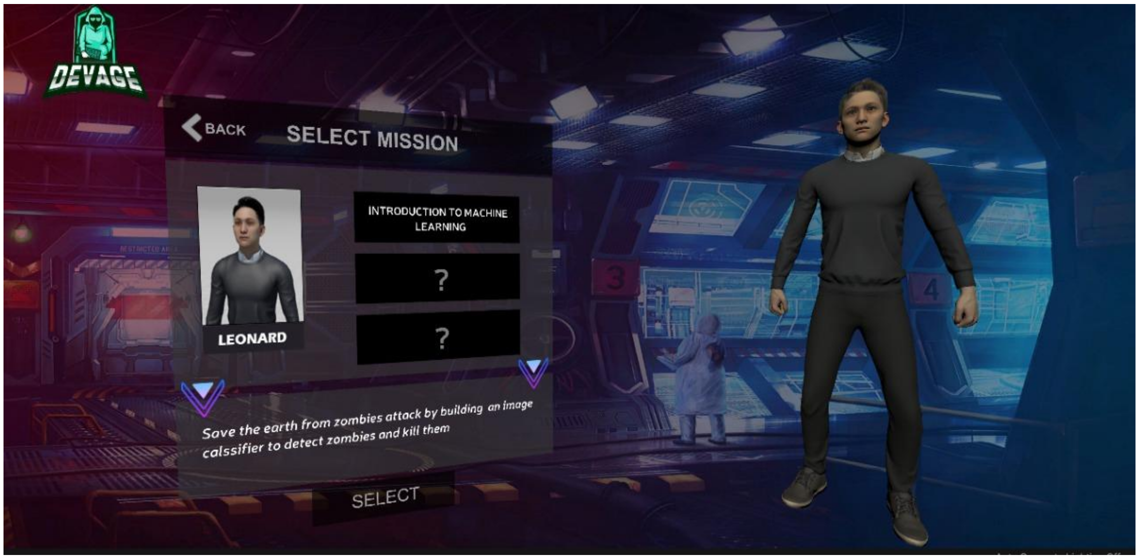


2. Choose character to play with



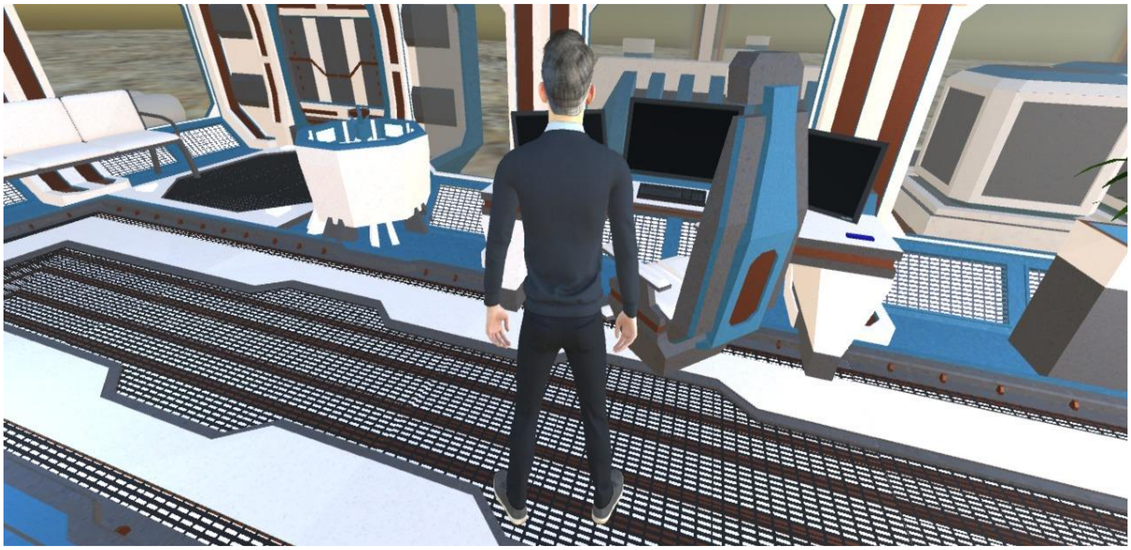
Leonard is a machine learning engineer that seeks to learn more about machine learning

3. Choose the mission



Save the earth from zombies attack by building an image calssifier to detect zombies and kill them

4. Start playing!



Character actions controls:

W: forward

S: Backward

D: Right

A: Left

F: Pick

V: Turn on PC

C: Change camera mode

5. Installation Guide:

1. Install python 3 with tensorflow
2. Make the installation path of python in C:/Programs
3. Install game.exe

Conclusion and Future Work

Conclusion

This project is a good model to be used in the educational organizations as it follows a completely new approach in the technological education, which is game-based learning. The idea is scalable, and it can include a lot of other topics and learning outcomes. The strengths of this project lie in creating an appropriate story for each educational topic that clearly achieves the desired learning outcomes.

The visual aspect also is a very important factor in this project because the game should attract the player to be motivated to learn with some entertainment. So, it has been designed carefully to meet these criteria. Studying the user experience is an essential part of the designing process. It has been designed to be as near as possible from the predicted visuals by the intended users.

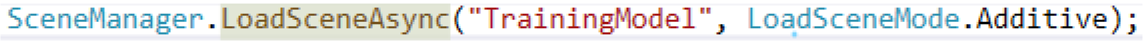
As a result of testing the game on real students from computer science major, 90% of the students have given a positive feedback about the game-based learning and after a simple survey, 85% of them have clearly understood the desired learning outcomes. They suggested to add more challenges in the game and add new levels with a related story among all the levels.

Future Work

As planned before, we have implemented the short vision of the project, which is developing the first mission of level 1 (Image Classification). For the long vision, there will be graphics enhancements with adding some challenging obstacles to the game levels. There will be also other new levels concerning the basic concepts of programming such as algorithms, data structures, etc...

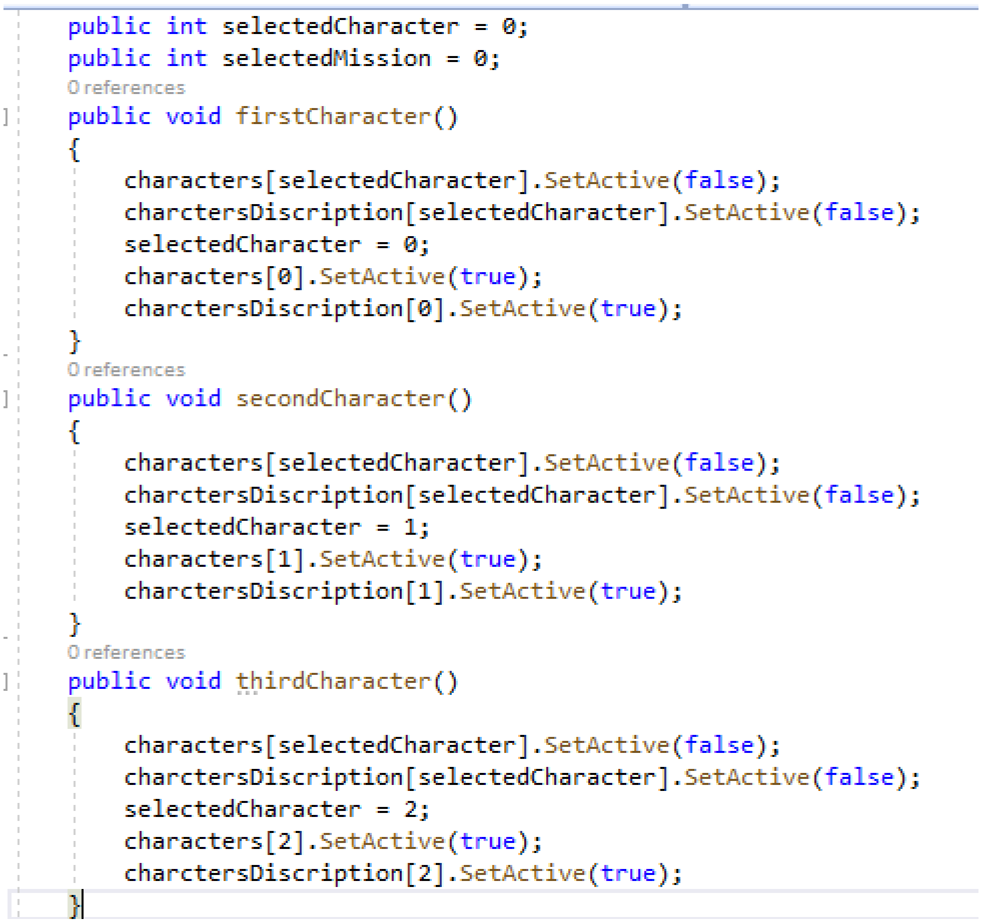
The game levels will have a related story and each level will have its challenging target that should be accomplished to be able to move to the next level. The game also may include other fields such as medical fields by simulating the surgeries and experiments.

Appendices



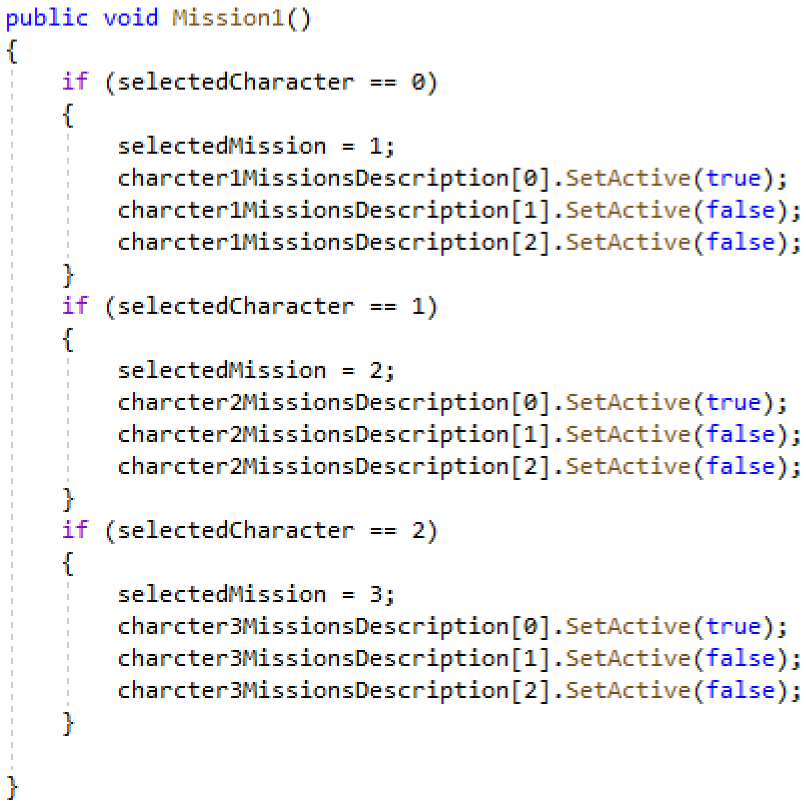
```
SceneManager.LoadSceneAsync("TrainingModel", LoadSceneMode.Additive);
```

Appendix 1



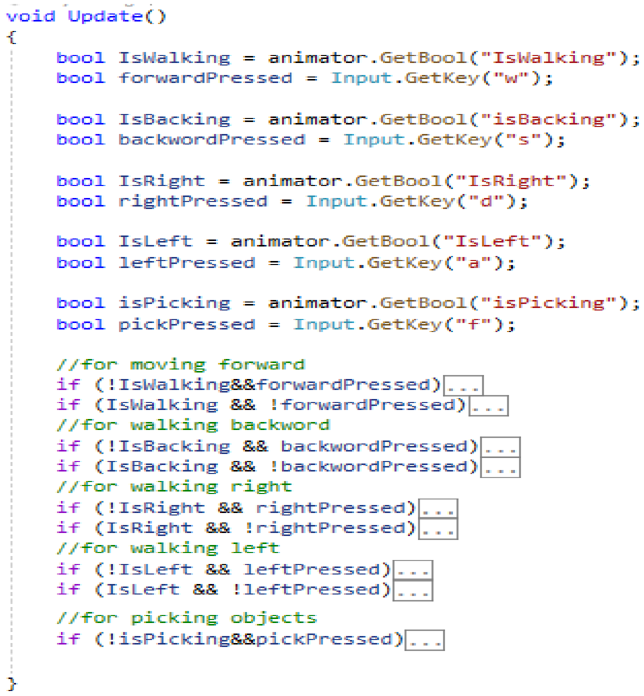
```
public int selectedCharacter = 0;
public int selectedMission = 0;
0 references
public void firstCharacter()
{
    characters[selectedCharacter].SetActive(false);
    charctersDiscription[selectedCharacter].SetActive(false);
    selectedCharacter = 0;
    characters[0].SetActive(true);
    charctersDiscription[0].SetActive(true);
}
0 references
public void secondCharacter()
{
    characters[selectedCharacter].SetActive(false);
    charctersDiscription[selectedCharacter].SetActive(false);
    selectedCharacter = 1;
    characters[1].SetActive(true);
    charctersDiscription[1].SetActive(true);
}
0 references
public void thirdCharacter()
{
    characters[selectedCharacter].SetActive(false);
    charctersDiscription[selectedCharacter].SetActive(false);
    selectedCharacter = 2;
    characters[2].SetActive(true);
    charctersDiscription[2].SetActive(true);
}
```

Appendix 2



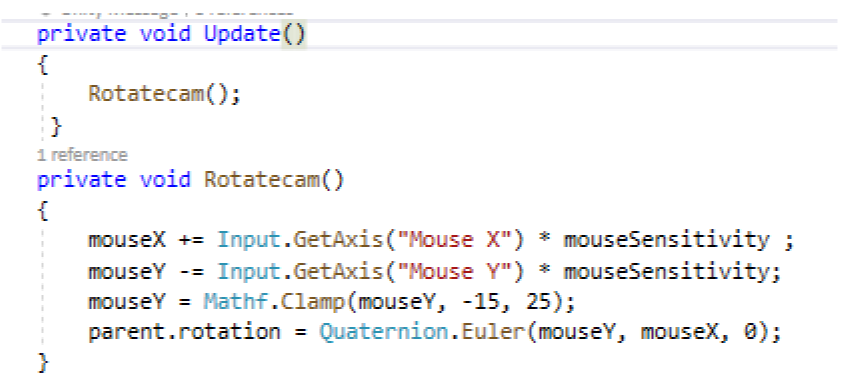
```
public void Mission1()
{
    if (selectedCharacter == 0)
    {
        selectedMission = 1;
        charcter1MissionsDescription[0].SetActive(true);
        charcter1MissionsDescription[1].SetActive(false);
        charcter1MissionsDescription[2].SetActive(false);
    }
    if (selectedCharacter == 1)
    {
        selectedMission = 2;
        charcter2MissionsDescription[0].SetActive(true);
        charcter2MissionsDescription[1].SetActive(false);
        charcter2MissionsDescription[2].SetActive(false);
    }
    if (selectedCharacter == 2)
    {
        selectedMission = 3;
        charcter3MissionsDescription[0].SetActive(true);
        charcter3MissionsDescription[1].SetActive(false);
        charcter3MissionsDescription[2].SetActive(false);
    }

}
```

Appendix 3



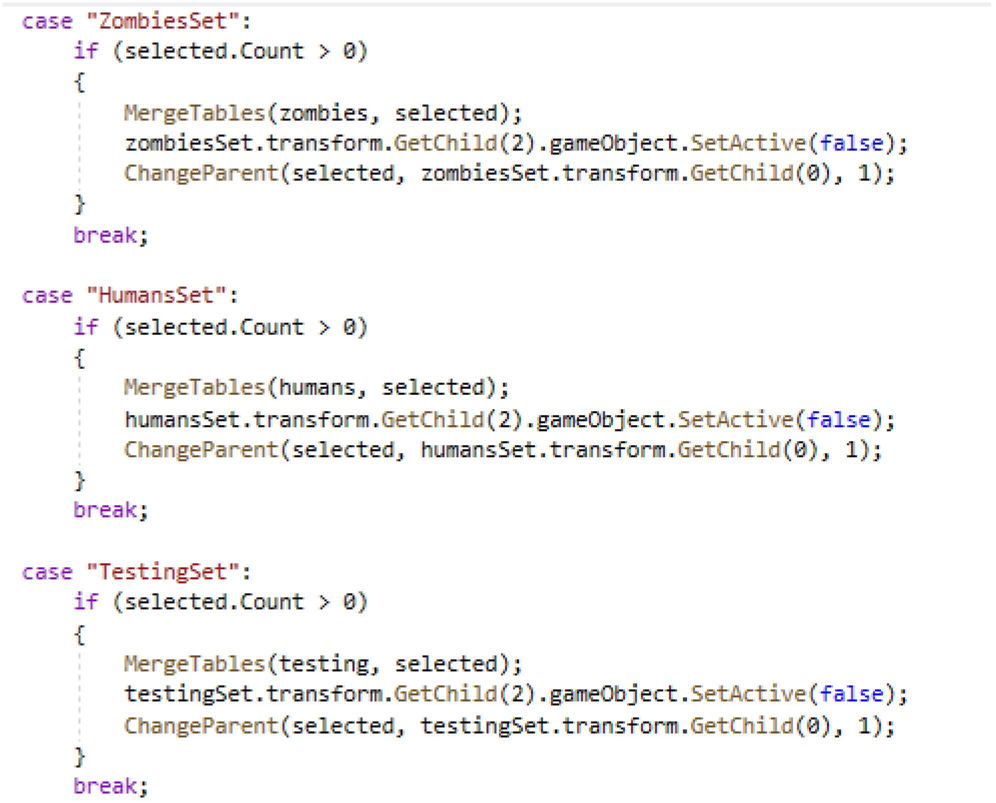
```
void Update()
{
    bool IsWalking = animator.GetBool("IsWalking");
    bool forwardPressed = Input.GetKey("w");

    bool IsBacking = animator.GetBool("isBacking");
    bool backwordPressed = Input.GetKey("s");

    bool IsRight = animator.GetBool("IsRight");
    bool rightPressed = Input.GetKey("d");

    bool IsLeft = animator.GetBool("IsLeft");
    bool leftPressed = Input.GetKey("a");

    bool isPicking = animator.GetBool("isPicking");
    bool pickPressed = Input.GetKey("f");

    //for moving forward
    if (!IsWalking&&forwardPressed)...
    if (IsWalking && !forwardPressed)...
    //for walking backword
    if (!IsBacking && backwordPressed)...
    if (IsBacking && !backwordPressed)...
    //for walking right
    if (!IsRight && rightPressed)...
    if (IsRight && !rightPressed)...
    //for walking left
    if (!IsLeft && leftPressed)...
    if (IsLeft && !leftPressed)...
    //for picking objects
    if (!isPicking&&pickPressed)...

}
```

Appendix 4



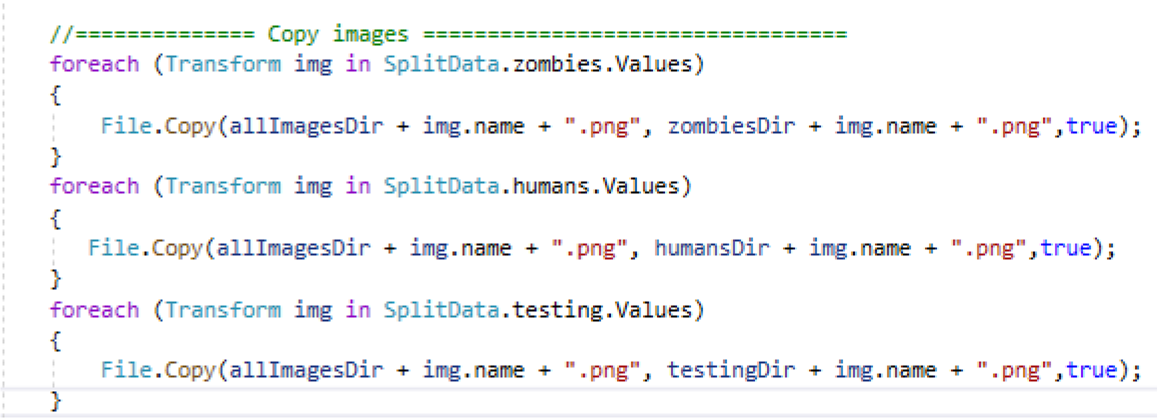
```
private void Update()
{
    Rotatecam();
}
1 reference
private void Rotatecam()
{
    mouseX += Input.GetAxis("Mouse X") * mouseSensitivity ;
    mouseY -= Input.GetAxis("Mouse Y") * mouseSensitivity;
    mouseY = Mathf.Clamp(mouseY, -15, 25);
    parent.rotation = Quaternion.Euler(mouseY, mouseX, 0);
}
```

Appendix 5



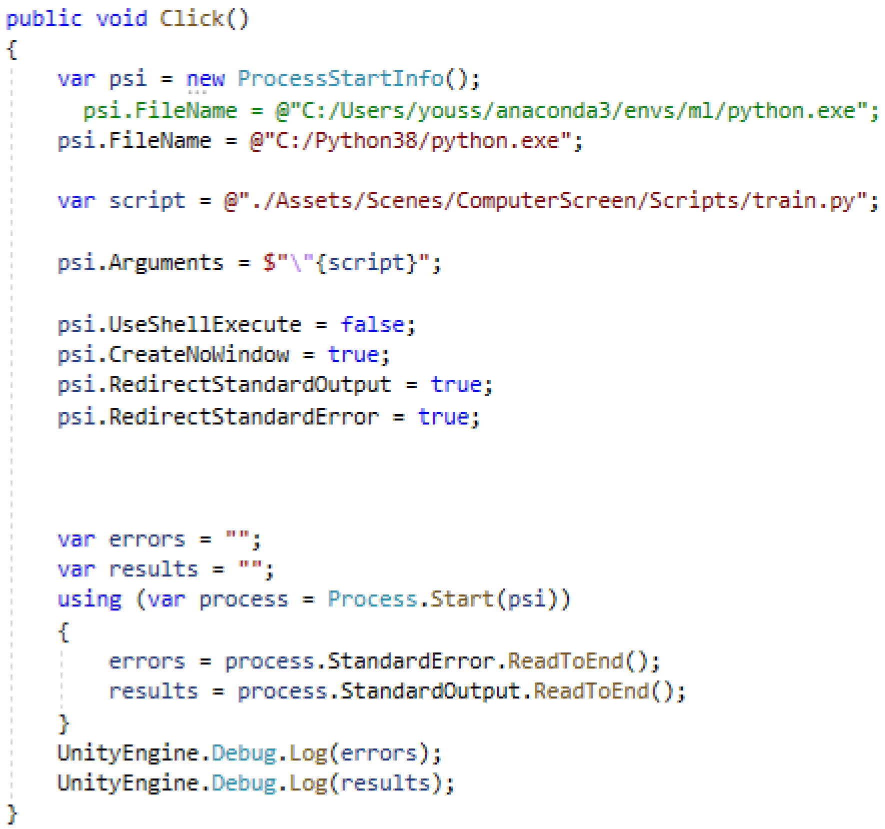
```
case "ZombiesSet":
    if (selected.Count > 0)
    {
        MergeTables(zombies, selected);
        zombiesSet.transform.GetChild(2).gameObject.SetActive(false);
        ChangeParent(selected, zombiesSet.transform.GetChild(0), 1);
    }
    break;

case "HumansSet":
    if (selected.Count > 0)
    {
        MergeTables(humans, selected);
        humansSet.transform.GetChild(2).gameObject.SetActive(false);
        ChangeParent(selected, humansSet.transform.GetChild(0), 1);
    }
    break;

case "TestingSet":
    if (selected.Count > 0)
    {
        MergeTables(testing, selected);
        testingSet.transform.GetChild(2).gameObject.SetActive(false);
        ChangeParent(selected, testingSet.transform.GetChild(0), 1);
    }
    break;
```

Appendix 6



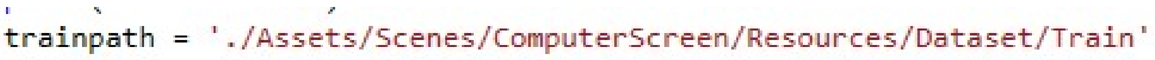
```
//============== Copy images ==================================
foreach (Transform img in SplitData.zombies.Values)
{
    File.Copy(allImagesDir + img.name + ".png", zombiesDir + img.name + ".png",true);
}
foreach (Transform img in SplitData.humans.Values)
{
   File.Copy(allImagesDir + img.name + ".png", humansDir + img.name + ".png",true);
}
foreach (Transform img in SplitData.testing.Values)
{
    File.Copy(allImagesDir + img.name + ".png", testingDir + img.name + ".png",true);
}
```

Appendix 7



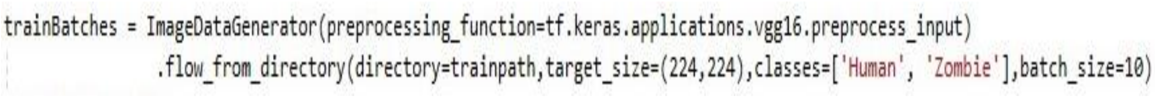
```
public void Click()
{
    var psi = new ProcessStartInfo();
      psi.FileName = @"C:/Users/youss/anaconda3/envs/ml/python.exe";
    psi.FileName = @"C:/Python38/python.exe";

    var script = @"./Assets/Scenes/ComputerScreen/Scripts/train.py";

    psi.Arguments = $"\"{script}";

    psi.UseShellExecute = false;
    psi.CreateNoWindow = true;
    psi.RedirectStandardOutput = true;
    psi.RedirectStandardError = true;



    var errors = "";
    var results = "";
    using (var process = Process.Start(psi))
    {
        errors = process.StandardError.ReadToEnd();
        results = process.StandardOutput.ReadToEnd();
    }
    UnityEngine.Debug.Log(errors);
    UnityEngine.Debug.Log(results);
}
```

Appendix 8



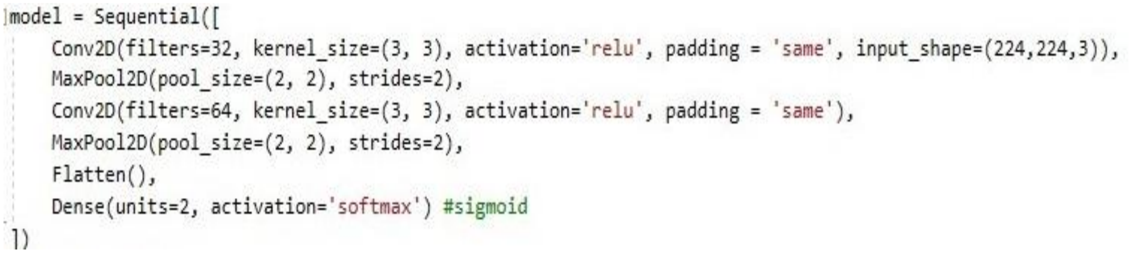
```
trainpath = './Assets/Scenes/ComputerScreen/Resources/Dataset/Train'
```

Appendix 9



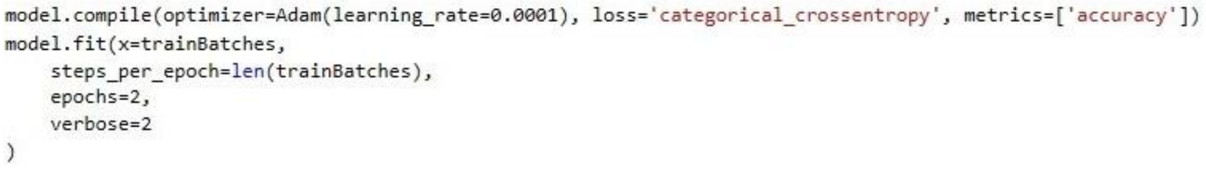
```
trainBatches = ImageDataGenerator(preprocessing_function=tf.keras.applications.vgg16.preprocess_input)
            .flow_from_directory(directory=trainpath,target_size=(224,224),classes=['Human', 'Zombie'],batch_size=10)
```

Appendix 10



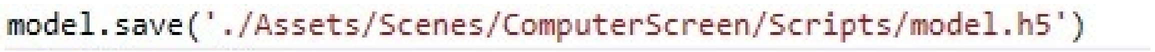
```
model = Sequential([
    Conv2D(filters=32, kernel_size=(3, 3), activation='relu', padding = 'same', input_shape=(224,224,3)),
    MaxPool2D(pool_size=(2, 2), strides=2),
    Conv2D(filters=64, kernel_size=(3, 3), activation='relu', padding = 'same'),
    MaxPool2D(pool_size=(2, 2), strides=2),
    Flatten(),
    Dense(units=2, activation='softmax') #sigmoid
])
```

Appendix 11



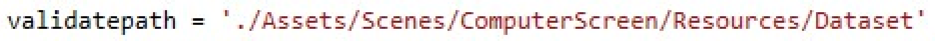
```
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x=trainBatches,
    steps_per_epoch=len(trainBatches),
    epochs=2,
    verbose=2
)
```

Appendix 12



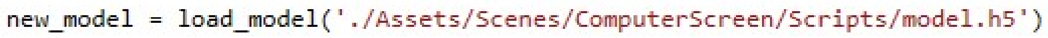
```
model.save('./Assets/Scenes/ComputerScreen/Scripts/model.h5')
```

Appendix 13



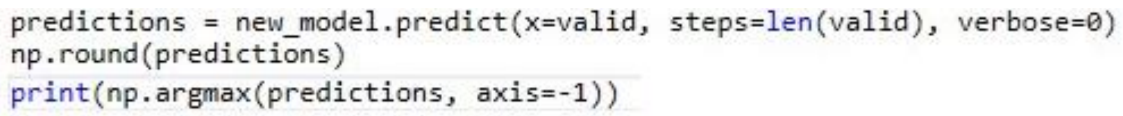
```
validatepath = './Assets/Scenes/ComputerScreen/Resources/Dataset'
```

Appendix 14



```
new_model = load_model('./Assets/Scenes/ComputerScreen/Scripts/model.h5')
```

Appendix 15



```
predictions = new_model.predict(x=valid, steps=len(valid), verbose=0)
np.round(predictions)
print(np.argmax(predictions, axis=-1))
```

Appendix 16

References

1. What’s Better: Innovative or Traditional Teaching Methods (unicheck.com).
   Retrieved from: https://unicheck.com/blog/traditional-innovative-teaching
   Last accessed: November 16, 2020.

2. De Freitas, S. (2018). **Are games effective learning tools? A review of educational games**. Journal of Educational Technology & Society, 21(2), 74-84.

3. Kiili, K., De Freitas, S., Arnab, S., & Lainema, T. (2012). **The design principles for flow experience in educational games**. Procedia Computer Science, 15, 78-91.

4. Computer Science for Kids Blog | CodeMonkey.
   Retrieved from: https://www.codemonkey.com/blog
   Last accessed: November 11, 2020.

5. 6 Deep Learning models — When should you use them? | by Rohan Gupta | Towards Data Science.
   Retrieved from: https://towardsdatascience.com/6-deep-learning-models-10d20afec175
   Last accessed: November 17, 2020.

6. [Yousef Ahmed]. (2019, May 14). Play Code Demo - YouTube.
   Retrieved from: https://www.youtube.com/watch?v=xD1IbbCq444&feature=youtu.be

7. Harnessing the power of game dynamics1: Why, how to, and how not to gamify the library experience | Bohyun Kim | College & Research Libraries News (acrl.org).
   Retrieved from: https://crln.acrl.org/index.php/crlnews/article/view/8811/9409
   Last accessed: November 16, 2020.

8. What is GBL (Game-Based Learning)? | Editorial Team | EdTechReview
   Retrieved from: https://edtechreview.in/dictionary/298-what-is-game-based-learning#:~:text=Game%20based%20learning%20(GBL)%20is,matter%20to%20the%20real%20world.
   Last accessed: November 20, 2020.

9. Why Game-Based Learning? | the Learning Counsel
   Retrieved from: https://thelearningcounsel.com/article/why-game-based-learning
   Last accessed: November 20, 2020.

10. The theory of game-based learning | Gamelearn Team | (game-learn.com)
    Retrieved from: https://www.game-learn.com/the-theory-of-game-based-learning
    Last accessed: November 21, 2020.

11. 4 Stunning Examples of Game Based Learning | Arjun Sanal | (playxlpro.com)
    Retrieved from: https://playxlpro.com/four-key-examples-of-game-based-online-learning
    Last accessed: November 21, 2020.

12. R. Van Eck, “Building artificially intelligent learning games,” in Games and Simulations in Online Learning, D. Gibson, C. Aldrich, and M. Prensky, Eds. Hershey, PA: Idea Group, 2006, pp. 271-307.
13. R.L. Appelman, “Serious game design: Balancing cognitive and affective engagement”, Proc. ISAGA, 2007.