Fall

2022-2023

Games Engine and Scripting

Name: Mahmoud Rumaneh

ID: 20120103

Date: February 2, 2023

Eng. Daina Al-Said

Contents

https://youtu.be/L7zLUkh2zt4

Part 1

P1

1. Unity: Unity is a widely used game engine and development platform developed by Unity Technologies, a Danish company, and first released in 2005. Initially, the engine was mainly used for game development for Mac desktop computers but has since been expanded for Windows to support multiple platforms. Unity Technologies constantly releases new versions of the engine, each with new features, improvements, and performance improvements. The major update was the release of Unity 4 in 2012, which allowed mobile game development, and Unity 5 in 2015, which introduced a new physics-based rendering engine, among other things. Unity has become a popular choice among indie game developers due to its cost-effectiveness and ease of use. This has created a large community of developers and users who share resources and knowledge. In recent years, Unity has been used to create many successful games such as Monument Valley, Pokemon Go, and Angry Birds 2. In 2017, Unity Technologies announced that the engine has been used to create more than half of all new mobile games. It has also been used to create various types of interactive content such as architectural visualizations, training simulations, and interactive product demos. Recently, Unity has been actively developing and expanding its capabilities for creating virtual and augmented reality applications. In 2020, Unity announced the release of Unity 2020, which included major updates for XR development such as the introduction of one-pass instance rendering technology, which improves the performance of virtual reality (VR) applications. [1]

Unity has had a significant impact on game design by making it easier for developers to create and publish games for multiple platforms. Unity supports a variety of platforms, including PC, mobile, console, and virtual reality/augmented reality devices, allowing developers to reach a larger audience and expand the potential market for their games. Another way Unity has influenced game development is by making it easier to create and implement game assets like 3D models, textures, and animations. Unity's asset store has a large selection of ready-made assets that can be easily imported and integrated into a game, saving developers time and resources. Unity also makes it easier for developers to create interactive and immersive experiences by providing a variety of physics, animation, and scripting tools. As a result, the number of games with realistic physics and engaging gameplay mechanics has increased. [2]

2. Unreal: Epic Games created the Unreal Engine, which debuted in the 1998 first-person shooter game Unreal. The engine was created to be flexible and modifiable, allowing developers to create a wide range of games with ease. Many critically acclaimed and commercially successful games have been created using the engine over the years, including first-person shooters, action-adventure games, and sports games. This game engine is widely used by game developers as a tool in their work. Despite the fact that Unity is used by the vast majority of game developers. It offers fundamental development techniques that are extremely beneficial to newcomers and beginners. The engine has also been used in a variety of non-gaming applications, such as architectural visualization and film production. In recent years, the engine has become increasingly popular for virtual reality development. [3]

Unreal Engine has had a significant impact on game design by providing advanced graphics and rendering capabilities. With realistic lighting, materials, and particle effects, the engine has been used to create some of the most visually impressive games on the market. This has raised the bar for game visual quality and pushed other game engines to provide similar capabilities. [4]

Unreal Engine has influenced game development by introducing Blueprints, a visual scripting system. This system enables developers to quickly prototype and build gameplay mechanics without having to write code, which can be time-consuming and complex. This has simplified the creation of interactive and engaging gameplay experiences for developers. [4]

The Unreal Engine community is large and active, and it offers a wide range of tutorials, assets, and plugins that can be easily integrated into a game, saving developers time and resources. This has increased the diversity and creativity in the games industry by making game development more accessible to a wider range of people, including hobbyists and indie developers. [4]

Unity Vs Unreal: [5]

Versatility: [5]

- **3D:** Both engines have awesome 3D features, but Unreal is the best in terms of graphical fidelity.
- **2D:** Both engines have 2D, but Unity has a much larger focus and larger tools features.
- **Virtual Reality (VR):** Unity performs better in VR because the plugins are more versatile and integrate into the overall XR infrastructure.
- **Augmented Reality (AR):** Both engines support AR, but Unity has been doing it for a longer time and has more defined systems.
- **Multiplayer:** Both engines support multiplayer, but Unreal is the only one that has it built in. Although there are many third-party frameworks, Unity's integrated multiplayer is still in development.
- **Mobile:** Unity is the best engine for mobile.

Coding: [5]

In Unity, it writes the code in C#, whereas in Unreal, it uses C++. C++ is a more difficult language to learn, though Unreal has its built-in visual scripter called Blueprints. Visual scripting is a great alternative to coding because it allows you to do the same things as coding but without the need for coding. Simply create nodes and connect them to develop logic for your game.

While both engines support visual scripting, Unreal Engine's Blueprints Visual Scripting system has been around longer and is a more well-established method of "coding" for the engine. Though recent versions of Unity have added visual scripting as an option, C# programming is still considered the standard method of scripting behaviors.

Industry Presence: [5]

The most popular engine for indie developers and mobile games is Unity. Unity is used to create a variety of larger games, including Ori, Rust, Hearthstone, Angry Birds, and the majority of mobile games.

Unreal is more popular in the AAA industry than Unity. The engine is used in games such as Fortnite, Sea of Thieves, Star Wars, and many others.

Apart from small educational projects, Unity does not create their own games. Epic Games (developers of the Unreal Engine) on the other hand, has used the Unreal Engine to create many games such as Fortnite and Gears of War.

Community: [5]

Both engines have a sizable online presence, with their forums, Sub-Reddits, YouTube channels, and so on.

Unity: Unite is a yearly game developer convention. The majority of game development YouTubers concentrate on using and teaching Unity.

Unreal: Epic Games has a stronger online presence, including live tutorials.

Both engines have asset stores of their own. An asset store is a marketplace for 3D models, textures, systems, and other items for an engine that can be obtained for free or at a cost. These can be useful for developers who are not the best artists or who lack knowledge in a specific area.

M1

The origin of games engines: Games engines are software frameworks that provide a set of tools and technologies for developing video games. They are also known as game engines or simply engines. They typically include rendering, physics, audio, and scripting components, among other features. Game engines date back to the early days of video game development in the 1970s and 1980s, when developers had to write all of the necessary code from scratch for each game. However, as the industry matured and games became more sophisticated, it became clear that a more efficient approach was required. [6]

The "DOOM engine," created by id Software for their hit game DOOM in 1993, was one of the first widely used game engines. This engine, along with others such as the Quake and Unreal engines, paved the way for modern game engines. [6]

ID Software issued a press release prior to the release of their next game DOOM in 1993. DOOM was supposed to push the boundaries of what was thought possible on PCs. It summed up the amazing innovations in technology, gameplay, distribution, and content creation. He also coined a new term, (DOOM Engine). This term refers to the technology under the hood of the latest driver game drivers. The press release promised a new type of (open game), and ID game engine technology became the driving force behind the new computer game industry. These analyses support the conclusion that the invention of this game technology was a discrete historical event in the early 1990s. [6]

Types of games engines: [7]

1. 2D Game Engines: These engines are specifically designed for the creation of 2D games. They usually include tools for creating and manipulating 2D sprites, as well as tools for handling collision detection and physics in 2D environments. GameMaker Studio, Construct, and Stencyl are examples of popular 2D game engines. [7]

In mathematics and games, two-dimensional refers to a two-dimensional flat image. When discussing 2D games, many people are surprised to learn that it's not just perspective, especially when looking at popular side-scrolling games like Mario and Sonic. For example, "Bionic Commando Rearmed" is a side-scroller with 3D graphics to add depth to the game, so it was made in a 3D engine but with a side-scroller view. Aside from the obvious difference that 2D engines can only render 2D graphics, 3D and 2D engines are very similar in terms of functionality. Perspective refers to the ability to create objects that appear three-dimensional but are two-dimensional. [7]

2. 3D Game Engines: These engines are intended for 3D game development and include tools for creating and manipulating 3D models, as well as handling physics and lighting in 3D environments. Unity, Unreal Engine, and CryEngine are examples of popular 3D game engines. [7]

In 3D engines, they provide 3D representations of geometric data from the engine, which are mainly used for calculations and also for rendering 2D images. They are more complex than 2D engines because they must not only render 3D models but also apply 2D textures. Eventually, all 3D engines will have their physics engine, but there is the option to hire "average tools" to help assist a section of your game that you

wish to improve by using another engine. Moreover, 3D engines that do not require any programming are extremely rare, while most 2D engines use drag-and-drop features. [7]

3. Mobile Game Engines: These engines are specifically designed for creating games for mobile devices. Touch controls and support for different screen resolutions are typical features. Unity, Unreal Engine, and Marmalade, are some popular mobile game engines. [7]

Marmalade, formerly known as Airplay, is the Ideaworks3D Software Development Kit (SDK). It is a cross-platform suite for all mobile operating systems that includes everything needed to test and release mobile applications. The Marmalade engine is divided into two layers based on the API (Application Programming Interface) specification in the form of programming languages. One of the layers is "C", which is an abstraction layer that allows the user to access various device functions such as networking, memory management, audio and input methods, and so on. The other class, "C++", focuses on the development of 2D and 3D visual styles, with an emphasis on bitmap processing and grid rendering. This layer also includes the HTTP network, which may aid in the digital distribution of the game and its contents. [7]

4. VR/AR Game Engines: These engines are specifically designed for developing virtual reality (VR) and augmented reality (AR) games. They include features like head-tracking and VR/AR controller support. Unity, Unreal Engine, and CryEngine are some popular VR/AR game engines. [7]

5. Multiplayer Game Engines: These engines are specifically designed for creating multiplayer games. They include networking, matchmaking, and server management features. Unity, Unreal Engine, and Photon Engine are some popular multiplayer game engines. [7]

- **Associating game engines with influential titles that had a significant impact on video game design and development can reveal how these engines influenced player expectations over time. for instance:**

  a. As previously stated, the DOOM engine, created by id Software for the game DOOM in 1993, was one of the first widely used game engines. It was well-known for its fast-paced first-person shooter gameplay as well as its high-quality 3D graphics. This engine influenced the development of first-person shooter games, raising player expectations for immersive, fast-paced gameplay and high-quality graphics. [8]

  b. The Unreal Engine has been used in a number of notable games, including Unreal Tournament and Gears of War. These games are well-known for their excellent graphics, realistic physics, and intense multiplayer gameplay. The Unreal Engine has raised player expectations for immersive and visually stunning gameplay experiences by setting a high standard for visual quality and realism in games. [8]

  c. The Unity engine has appeared in numerous games, including Angry Birds 2 and Pokemon Go. These games demonstrated the engine's ability to create 2D and 3D games for mobile devices, as well as raising expectations for the quality and accessibility of mobile games. [8]

  d. CryEngine, created by Crytek, has been used in a number of influential games, including Far Cry and Crysis. These games have raised player expectations for immersive and visually stunning gameplay experiences due to their stunning graphics and realistic physics. [8]

M2

Fortnite [9] and the future of game engines are inextricably linked because Fortnite's success has helped to highlight the capabilities of Epic Games' Unreal Engine. The popularity of the game is also likely to have contributed to the engine's continued development and advancements. The game's success has been well documented, with over 350 million registered players as of 2021. It has also generated billions of dollars in

revenue, making it one of the most profitable games ever. Because of this success, has been a major driver of the growth of Unreal Engine, which is used by a wide range of games as well as non-gaming industries.

The use of new features in the game, such as real-time ray tracing and advanced AI, has set a new standard for the game industry. As a result, the number of games that use the Unreal Engine has increased, with many other game developers opting to use the engine to create high-quality graphics and smooth performance in their own games. Fortnite's free-to-play business model is also influencing the future of the game development industry, demonstrating that a free-to-play game can generate significant revenue. This has resulted in an increase in the number of other games adopting similar business models, and this trend is expected to continue in the future. [9]

The use of the Unreal Engine by Fortnite in non-gaming industries such as film and architecture is also helping to position the engine as a versatile tool for a variety of industries. This is expected to expand the engine's future opportunities and solidify its position as a leading game engine. [9]

For a variety of reasons, Fortnite could have a significant impact on **designers**, **developers**, and **gamers** in the future. Fortnite's success has highlighted the importance of creating engaging and immersive environments for **designers**. The use of vibrant colors, detailed textures, and interactive elements in the game has raised the bar for game design. As a result, designers may feel pressure to create similarly high-quality and engaging environments in their own games in order to meet Fortnite's expectations. [10]

Fortnite's success has shown **developers** the power of the free-to-play business model and the use of in-game purchases to generate significant revenue. As a result, developers may be more likely to adopt similar business models in the future, potentially altering how games are monetized. Furthermore, the game's use of advanced technology, such as real-time ray tracing and advanced AI, has set a new standard for game development, potentially putting developers under pressure to use similar technology in their own games. [11]

Fortnite has popularized the Battle Royale genre among **gamers**, resulting in an increase in the number of similar games being developed. This could result in more options for gamers in the future, as well as more competition among game developers to create the best Battle Royale experience. Furthermore, Fortnite's use of in-game purchases has resulted in a greater emphasis on in-game microtransactions, which may change how gamers interact with and purchase games in the future. [12]

D1

Games Engine: A gaming engine is a software development environment, also known as a (game architecture) or (game framework), that contains settings and configurations that optimize and simplify the development of video games in a variety of programming languages. A gaming engine may include a 2D or 3D graphics rendering engine that is compatible with various import formats, a physics engine that simulates real-world activities, artificial intelligence (AI) that automatically responds to the player's actions, a sound engine that controls sound effects, an animation engine, and a variety of other features. [13]

A game engine's primary purpose is to provide developers with a foundation on which to build their games, thereby reducing the amount of time and resources required to develop a game from scratch. This allows developers to concentrate on the game's distinguishing features, such as the story, characters, and gameplay mechanics. Unity, Unreal Engine, CryEngine, GameMaker Studio, and Godot are some examples of game engines. Each engine has its own set of advantages and disadvantages and can be used to create a variety of games. Some engines are better suited for 3D games than others for 2D games. Some are better suited to PC and console games, whereas others are better suited to mobile games. [13]

A number of factors can be considered when evaluating game engines, such as how well the engine runs on different hardware configurations and how efficiently it uses system resources. Furthermore, what tools and features, such as physics, animation, and scripting support, are included in the engine. Although, considered is the engine's usability, including documentation and the availability of tutorials and sample projects. The size and quality of the engine's community, as well as the availability of support and resources like tutorials and forums, are also important considerations. The engine's license cost and terms, including whether it is open-source or proprietary. The engine's ability to create games for multiple platforms, such as Windows, Mac, iOS, and Android. Finally, the best game engine for a specific project will be determined by the development team's specific needs and goals. [14]

PART 2

P2 & M3

Unity Features: [15]

1. 3D and 2D Graphics Support: Unity supports both 3D and 2D graphics, giving you the flexibility to choose the art style that best suits your projects. Each graphic type comes with its own specialized set of tools and script APIs to control different physics options that are optimized for the respective style. This allows you the freedom to choose the art style you want for your projects. Unity offers a robust set of tools for 3D graphics, including the ability to create custom materials, design shaders with the Shader Graph, customize lighting, apply post-processing effects, and much more. The engine also allows for the generation of 3D terrain and 2D tile maps, providing a comprehensive set of tools for any type of graphics used in your project. [15]

The 3D and 2D feature supported by Unity is very important to game designers and developers and affects them directly. Any game that is developed needs 2D and 3D graphics. The graphics, terrain, and maps increase the realism and aesthetics of the game. Unity provides a comprehensive set of tools and the ability to create custom textures that let designers and developers bring their imagination to life and emphasizes fostering Unity for creative thinking.

2. Unity Scripting API: Unity features a comprehensive scripting API that offers easy access to the most commonly used functions. This includes both general game features and specific API calls that allow you to access specific engine features and nuances. For instance, the Scripting API allows you to change the color of the text in the engine by code, just as you would do it via the Unity Inspector. This applies to all elements that can be accessed via the Unity Inspector, including position, rotation, materials, audio playback, and more. [15]

Also, providing Unity for the API feature is a very important thing for any game engine, which provides easy access to all functions in several ways, the scripting, for example, makes you able to control the game better and add touches to the developer, because through the script he can move the game as he wants, add and delete elements, turning certain elements on and off as well, and accessing elements within Unity.

3. Easy-to-Understand Architecture: Unity offers a straightforward approach to building the game architecture. In a Unity game project, each level is divided into Scenes, and each Scene contains all the necessary game objects for the player to interact with, such as the background, player character, enemies, bullets, etc. Unity also allows for parent-child relationships between objects in the Hierarchy, making it easy to add multiple objects to a single parent, like a player character. Additionally, Unity provides the Inspector tool, which provides quick access to all the properties of an object, allowing you to make changes without the need to constantly delve into the code. [15]

Unity's game approach is easy to understand and very easy to handle and any age group can learn to use it and develop masterful games, I see when kids are taught how to make games it's done on Unity from step one to last, Unity breaks down each level into scenes, each scene contains all the necessary elements for the player to interact with and the relationships between objects have a clear hierarchy, this arrangement makes it easier for a Unity user to create their game.

4. Cross-platform Build Support: Unity games can be exported to a wide variety of platforms, including Android, iOS, Windows, MacOS, Linux, PS4, Xbox One, and more by downloading the appropriate kit. Additionally, you can also export HTML5 games if you want to publish your game on the web. The engine also minimizes the need for changes to the project for each platform, thus reducing the need for multiple versions of the project. [15]

Exporting games on several platforms is a very useful feature for anyone who wants to create a game that spreads quickly and is played by the largest possible number of people. Each platform has its fans and specific age groups, and the presence of the game on several platforms with the same quality provides great popularity for it and a huge number of downloads, so we see that most games Fame has been designed by Unity.

5. Large Asset Store: Unity offers a variety of resources for graphical assets, game genre templates, audio, particle effects, and more. Its extensive asset store includes a wide range of paid and free assets suitable for any game project. The assets are created by Unity and the community, providing a wide selection to choose from. Additionally, Unity makes it easy to add assets to your collection and install them into your project using the package manager, eliminating the need for manual file manipulation. [15]

Unity provides many useful resources for games such as graphical assets as well as sound, effects, etc., and this is very important for the game designer to find graphic assets easily and can be easily transferred to Unity as well and used, for example, in any game I wanted to design and needed assets, so the Unity assets store is my first choice using the package manager.

6. Virtual Reality (VR) & Augmented Reality (AR) Capabilities: Unity is a leading platform for developing VR and AR, despite their relatively new technologies. It offers various VR packages that support almost all VR headsets on the market and are frequently updated to adapt to the changing technology. You can also run your VR games through the Unity engine. Unity also provides ARCore and ARKit packages for AR development, as well as AR Foundation, which enables Unity developers to create AR apps for both Android and iOS simultaneously, eliminating the need for separate projects. [15]

VR and AR have become the present and the future. It is a very beautiful thing as a game designer to see the games engine that you use that supports AR and VR. Unity supports many packages related to VR and is frequently updated and keeps pace with continuous developments. It is an important thing for the player because the generation's next, focus will be on VR games.

7. Rendering Pipeline Options: Rendering graphics on a computer can be a complex process, and the way it is done can greatly affect the performance of your game. To address this, Unity offers a variety of built-in render pipelines for getting your game from the scene to the screen. This gives developers the flexibility to choose the pipeline that best fits their project's graphics needs. Moreover, Unity provides the Scriptable Render Pipeline API, allowing developers to create their own custom pipeline if needed. [15]

Because rendering graphics to game designers is difficult, it is very useful to see Unity provide an easy way to use and control graphics through built-in pipelines, also we can create our own pipeline.

8. Unity Developed Packages: Unity offers a wide range of free in-house developed packages and assets that enhance the functionality of the engine. These include the Bolt asset, which enables visual scripting within Unity, and Unity Playground, a 2D game framework for learning game development without coding from scratch. Additionally, Unity provides a variety of free models and game kits, allowing for easy access to Unity-approved assets for practice and experimentation. [15]

It is very important for the designer to see examples of ready-made games in the form of packages provided by Unity, which makes the game designer learn more quickly and see how the game is composed practically and without coding. Free games provide an excellent way to practice and experiment.

9. Animation Tools: Unity offers a comprehensive set of animation tools that cater to both 3D and 2D graphics. While it is possible to import animations from external programs such as Blender, Unity also enables you to create animations within the engine itself. These tools include adjusting the position and rotation of entire objects, as well as physically manipulating the bones in 3D models. Additionally, Unity provides the ability to add bone rigging to 2D images. All these features are also accessible through the Scripting API, providing you with unparalleled control over how animations work. Furthermore, the Animator system allows for the creation of an animation state machine, enabling smooth transitions between animations based on player actions, such as jumping. The Animator is also presented in a visually intuitive graph style, making it easy to understand how everything is connected. [15]

Animation is an essential thing in any game that the designer wants to design, Unity offers a comprehensive set of 3D and 2D animation tools, and through my experience with Unity, I saw great ease in the way to use animation and put the physical points of the animation that I want to use and determine the extent and the speed of his movement and the steps that he will follow, as well as adding the script to control the animation better.

10. Adaptability to Other Industries: Unity has a diverse range of features that extend its usefulness beyond traditional game development. Its animation tools and pipeline options make it a popular choice for indie filmmakers to create high-quality CG films. Additionally, Unity has developed tools like Unity Reflect aid building developers in visualizing their projects and connecting them to other CAD software. [15]

Something I liked very much is that I see Unity is no longer concerned with games only, and that its features can be applied to something else, which is the world of filmmaking, with very high quality, using new tools developed by Unity. I saw some videos that Unity made and was impressed with the accuracy of the visual images, and I see that Unity has a great future in the film industry.

Unreal Features: [16]

1. Python scripting: Unreal Engine allows for integration into your pipeline and automation of workflows with Python scripting support in the Unreal Editor. This allows for the creation of asset management pipelines, automation of data preparation, procedural level design, and custom user interface controls for the Unreal Editor. [16]

It's nice to see Unreal use python as a language that can be used in developing games because we know how advanced and easy it is to use and how to write code in python, and Unreal gives many advantages that are in turn to python, which I call the language of the current era.

2. The Unreal Editor: The Unreal Engine package includes the Unreal Editor, a cross-platform integrated development environment for content creation and game level development. The Unreal Editor supports Multi-User Editing, allowing multiple team members to work on the same project simultaneously and safely.

Additionally, the editor can be run in VR mode, providing a WYSIWYG environment for building and designing. It is available on Linux, macOS, and Windows. [16]

Unreal also provides the Unreal editor, which provides designers with ease of editing because it supports multi-user editing, which provides ease for members of the same team to work on the same project, and this is a very useful thing that attracts any designer to use Unity.

3. Asset optimization: Optimizing a complex model for real-time performance can be a tedious process that requires multiple iterations. Unreal Engine provides tools to streamline this process, such as automatic generation of LOD, the Proxy Geometry tool for combining multiple meshes and materials into one, and various modeling tools for simplifying meshes and removing hidden polygons. These tools help to simplify and speed up the process of preparing a model for real-time use. [16]

We all suffered as designers when we use a certain asset, it is difficult for us to improve it as we want, Unreal has provided us with automatic optimization features for the assets we need and shows the assets in the best possible way by removing any hidden polygons, which speeds up the game that we want to design and create.

4. Sky, cloud, and environment lighting: Unreal Engine allows for the creation of realistic or stylized skies, clouds, and other atmospheric effects with a Volumetric Cloud component that can interact with the Sky Atmosphere, Sky Light, and up to two directional lights. The component can be lit and shadowed dynamically in real-time, corresponding to the time-of-day updates, providing full artistic freedom. [16]

Many of us are looking for a game engine that designs the game with complete realism, and this is what Unreal provides by providing the ability to create realistic skies, clouds, and anything realistic that you want to have in your game. Realism is what touches the player's feelings and brings him closer to the game more, also by making the game time the actual time of the player and a lot of features are completely realistic.

5. Animation Blueprints: Animation Blueprints allow for the creation and control of complex animation behaviors. It is a specialized Blueprint that manages the animation of a Skeletal Mesh. Graphs are edited within the Animation Blueprint Editor, where it's possible to blend animations, control bones of a Skeleton directly, or set up logic that defines the final animation pose for a Skeletal Mesh to use per frame. [16]

Unreal is also one of the game engines that provide ease in using the animation and modifying it per your game. For example, if the animation that you want to use is human, then you can modify the locations of joints and bones for this animation through Unity, which provides you with great realism and ease in your edit of animation.

6. Machine Learning Deformer: The Machine Learning (ML) Deformer generates high-quality simulations of nonlinear deformers, proprietary rigs, or any other type of deformation by training a machine learning model using a custom Maya plugin, which is then run in real-time in the Unreal Engine. This allows for the creation of film-quality deformations, such as flexing muscles, bulging veins, and sliding skin, that were previously difficult or impossible to achieve in real-time. [16]

The Machine Learning term has become very popular, and it is very nice to see Unreal use it for a high-quality simulation process for many things that you can choose for your game, for example, you can use it to cause distortions in the body that you want, which increases the realism of the game that you want to create and increases people’s love to play it.

7. Virtual Texturing: Unreal Engine provides two ways to support very large textures by dividing them into smaller tiles and loading only the visible tiles. Streaming Virtual Texturing, which uses Texel data from

converted textures stored on disk, reduces the memory overhead for light maps and detailed UDIM UV artist-created textures. Runtime Virtual Texturing, on the other hand, generates Texel data on the GPU during runtime, and improves rendering performance for procedural and layered materials. [16]

Texturing is an important thing in any game and it is sometimes difficult for the developer to deal with it. Unreal has provided streaming virtual texturing through which it improves the display of the texture within the game, which increases the purity and quality of the visual images and increases the quality of the game as a whole and ease of use as a better solution than the manual solution.

8. Movie Render Queue: Unreal Engine enables the creation of high-quality media for cinematics, marketing materials and linear entertainment without post-processing by allowing for the rendering of movies and stills with accumulated anti-aliasing and motion blur. The support for tiled rendering makes it possible to produce extremely high-resolution images, such as those intended for printed output. Additionally, the ability to output render passes enables downstream compositing. [16]

I see that there is a competition for game engines to allow filmmakers to make their films on gaming platforms, and because Unreal in particular provides great high quality, whether in games or in movies and has all the tools you need to produce a high-quality and elaborate movie.

9. Unreal Motion Graphics UI Designer: The UMG visual UI authoring tool allows for the creation of UI elements such as in-game/in-application HUDs, menus, or other interface-related graphics to be presented to users. Build your interface by using editable Blueprint Widgets for pre-made functions like buttons, checkboxes, sliders, and progress bars. [16]

The UI is one of the main elements of the game that shows the player how to use this game and what is the correct flow to use it. Without the UI, the player will not know in which step he is now and what he must do, and the UI adds a feature to the game and a special aesthetic character that attracts the player to play it.

10. Remote control protocols and web UI builder: Creating custom UIs that allow users to make settings from any device that has a web browser, including mobile devices, is made easy with a drag-and-drop UI builder and a fully REST-compliant API. You can even control lighting from LED volumes on set from an iPad. Additionally, support for OSC (Open Sound Control) protocol, an industry standard for bidirectional communication between various devices, sensors, and audio equipment is also provided. [16]

It is also very nice to see Unreal making the UI creation process easier and more so that you can only use your browser or use your smartphone using the drag-and-drop software, which added great ease to designers and stimulated their creativity.

P3

1. Temple Run: Imangi Studios, a small independent game studio, used Unity game engine to create Temple Run. Unity was employed to develop the game's 3D graphics, physics and user interface. Characters and environments were modeled and animated in 3D software like Maya or 3D Studio Max, then imported into Unity. The game's logic such as character movement, collision detection, and power-ups were created using Unity's scripting system. Additionally, audio and music for the game were also created in Unity. After the game was completed, it was exported to various mobile platforms such as iOS and Android for release. [17]

a. 3D and 2D Graphics Support: In creating Temple Run, the developers utilized Unity's support for both 3D and 2D graphics. 3D assets like characters and environments were modeled and animated in 3D software and imported into Unity for rendering in real-time. Meanwhile, 2D elements like UI elements

and certain textures were also incorporated to enhance the overall visual appeal of the game. The ability to seamlessly blend 3D and 2D elements within Unity allowed the developers to create a visually striking game. [17]

b. Rendering Pipeline Options: The developers of Temple Run likely utilized Unity's various rendering pipeline options, such as the Built-in Render Pipeline, Universal Render Pipeline, and High-Definition Render Pipeline to achieve the game's desired performance and visual requirements. The Built-in Render Pipeline is Unity's default and most compatible option, it is also the most flexible, allowing for customizations. Whereas, the Universal Render Pipeline and High-Definition Render Pipeline are more optimized and provide better performance but require more setup and may not be compatible with all systems. [17]

c. Animation Tools: Unity, a cross-platform game engine, offers a vast array of tools for animating characters and objects in games like Temple Run. These tools include the importation and animation of 3D models, the creation and editing of animations through the Unity animation editor, and the application of animations to in-game objects through Unity's animator component. Additionally, the engine provides scripting options for the creation of custom animation events and behaviors, making Unity's animation tools a robust and versatile choice for creating animations in Temple Run. [17]

2. Hitman Go: Hitman Go is a turn-based puzzle game that is a spin-off of the popular Hitman franchise. Developed by Square Enix Montreal, it was launched for mobile devices in 2014. Players take on the role of Agent 47 and move through challenging levels, using stealth, strategy, and various weapons, gadgets, and items to eliminate targets and finish objectives. The game is designed with a minimalist art style, and the levels are specifically created to be challenging puzzle experiences that test players' critical thinking and planning skills. The game was well-received by critics for its inventive and challenging gameplay, It developed by Unity Games Engine. [18]

a. Cross-platform Build Support: Unity, a cross-platform game engine, was likely used to develop Hitman Go, a mobile game. Unity offers built-in support for creating builds for several platforms, including iOS, Android, and Windows, allowing the developers to write the game code once and deploy it to multiple platforms without any significant changes. For instance, Unity can automatically handle different inputs like touch controls for mobile devices and keyboard and mouse controls for PC. Additionally, Unity has various built-in features to optimize game performance on various platforms, such as adjusting resolution and quality settings. Unity's cross-platform build support enables developers to easily create builds for multiple platforms, increasing the reach and player base of a game like Hitman Go. [18]

b. Animation Tools: Hitman Go used Unity's animation tools to create the animations of characters and objects in the game. Unity offers a variety of tools for animating, such as importing and animating 3D models, creating and editing animations using Unity's animation editor, and applying animations to game objects through Unity animator component. Unity also provides scripting capabilities which allows developers to create customized animation events and behaviors, providing more control over the animation of characters and objects in the game. Unity's animation tools provide a powerful and adaptable solution for creating the animations in Hitman Go. [18]

c. Unity scripting API: Unity's scripting API allows developers to write code in C#, UnityScript to control the behavior of objects in their game. In Hitman Go, developers utilized this API to access and modify game data such as player progress, game states, and level layout. They also used it to create custom events and triggers to control the flow of the game, like initiating a cutscene or ending a level

when specific conditions are met. Overall, Unity's scripting API offers a powerful and adaptable solution for creating custom logic and behavior in Hitman Go. [18]

3. Angry Birds: In Angry Birds, players use a slingshot to propel birds towards structures inhabited by pigs with the objective of eliminating all the pigs on the level. The game which was first introduced in 2009 has seen the release of multiple sequels and spin-offs and is available on a variety of platforms such as iOS, Android, and PC. It has gained immense popularity among mobile gamers. [19]

a. Animation Tools: The Unity game engine is utilized by Angry Birds to produce its animations and physics. The Unity physics engine is employed to replicate the realistic movements of the birds and pigs when they collide with structures or launched. Unity's animation tools such as the animator and animation controllers are also used to create animations for the birds and pigs, including their movements, expressions, and death animations. This gives the developer greater control over the characters, objects, and the movement and physics of the environment. [19]

b. Rendering Pipeline Options: Unity's default rendering option is the Unity Renderer, which supports a variety of platforms and devices and strikes a balance between performance and visual quality. The developer can select from various rendering pipelines provided by Unity, such as the lightweight Render Pipeline (LWRP) and the high-definition Render Pipeline (HDRP), to achieve the optimal combination of performance and visual quality for the game. In order to ensure a smooth performance on all devices, it is likely that the Angry Birds developers have employed the LWRP in the game. [19]

c. Large Asset Store: It is likely that the developers of Angry Birds utilized the Unity Asset Store to obtain a variety of pre-made assets for their game. The Unity Asset Store provides a wide selection of resources such as 3D models, animations, textures, and materials that can be used in various aspects of the game such as the birds and pigs, environments, and structures. The developers might have used these assets and tools from Unity Asset Store to create the game's unique animations, physics and special effects. [19]

M4

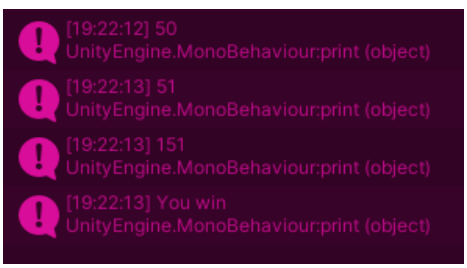
1. Roll a Ball: In roll a ball game we have a marble and 3 blue, red and green bottles, if the ball jumps over the boxes and takes the blue bottle the score will increase by 5 points and the red bottle will increase the score by 10 points and the green bottle will increase the score by 15 points then the ball must jump To skip all the boxes and collect these points and also collect the coins that each one of them increases the score by one point, and in the end if he reaches the end of the stage and takes the big bag of money, he will increase the score by a hundred and he will win the game.

Features:

1. I used the asset store for unity, to use the prefabs and textures, for all assets in the game.

2. I used the physics for the mass of the ball by the inspector in Unity.

3. I used the script by C# language to make the jump of the ball and the scoring.




```
void Update()
{
    hor = Input.GetAxis("Horizontal");
    ver = Input.GetAxis("Vertical");
    rb.AddForce(ver * -1, 0.6f, hor );
    if (Input.GetKeyDown(KeyCode.Space))
    {
        rb.AddForce(Vector2.up * jumpAmount, ForceMode.Impulse);
    }
```

2. Car and Boxes: The car and boxes game, its main objective is to see the physical ability of the car to displace the boxes and make the car move forward automatically, and that the street is renewed and destroyed automatically as well.

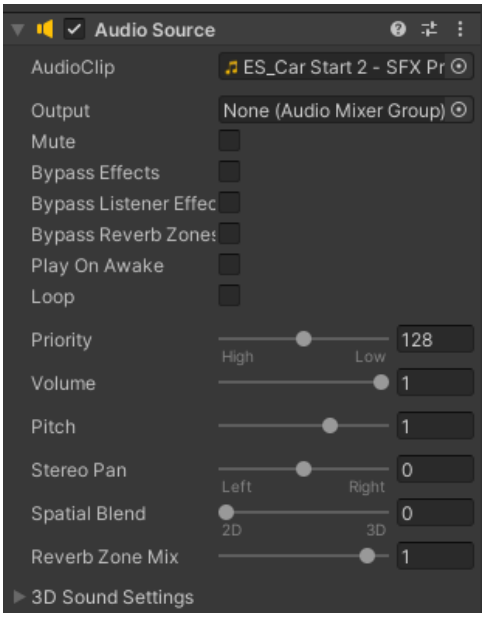
Features:

1. I used instantiation and destroying in this game.

2. I used the audio in this game, when the play start and the car touch the street, the audio will play.

3. I used the car, boxes, and street by the asset store in Unity.

4. I used the script for the instantiation and destroying, and also for the physics of the car and automatically going forward.



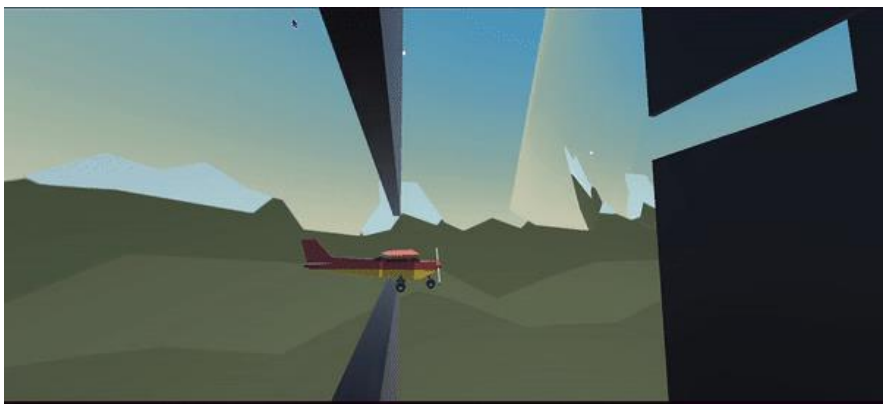
2. Plane & Barriers: The plane game is a game in which the plane is moved up and down only, and you must pass through these barriers, and on each barrier that is crossed for you a point, and when you take a yellow coin, it speeds up the plane, and if you take a red coin, it will slow the plane down, and this game does not end Except when you hit one of the barriers, it takes you back to the starting point.

Features:

1. I used instantiation and destroying for the barriers by the script.

2. I used the asset store for Unity for all assets such as the plane, barriers, coins, and background

3. I used the script for scoring and speeding and slowing the plane.



PART 3

P4

The game has been modified and new features have been added to make it better for the player.

PROJECT DESIGN DOCUMENT

5/1/2023

Mahmoud Rumaneh

PROJECT CONCEPT

1 **Player Control**	You control a	in this	
	Plane	*side view*	game
	where	makes the player	
	arrow keys (vertical)	*Move up and down*	
2 **Basic Gameplay**	During the game,		from
	Barriers and coins	appear	*The right side of the screen*
	and the goal of the game is to		
	Skip the barriers and get the coins		
3 **Sound & Effects**	There will be sound effects		and particle effects
	When the plane turns on		*when the plane takes the orange coin the speed will increase and it can be slowed by taking the red coin*

4 **Gameplay Mechanics**	As the game progresses,	making it
	The fan Is spinning	*Making the plane more realistic and flying*

5 **User Interface**	The	will	whenever
	score	*increase*	*when the plane skips the barrier, the score will increase by one*
	At the start of the game, the title		and the game will end when
	"Plane and Barriers"	will appear	*the plane collides with the barrier.*

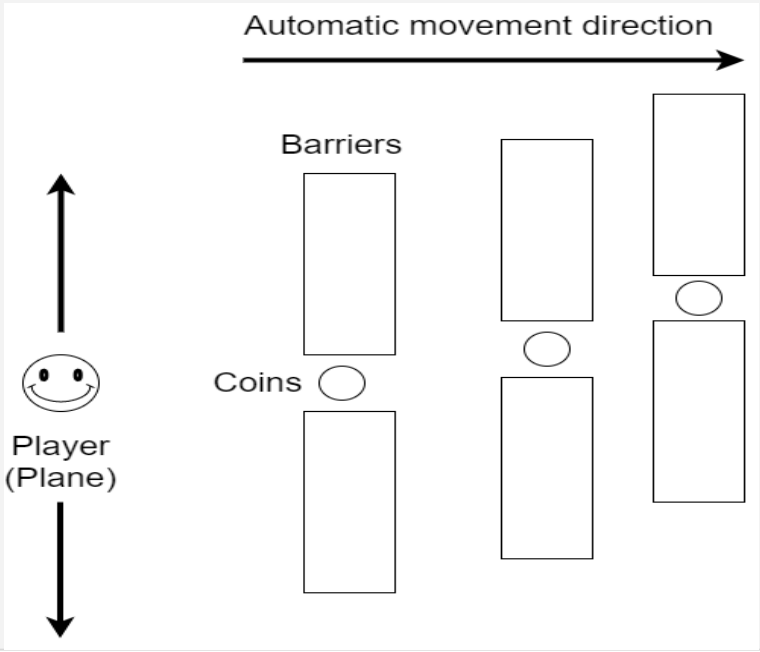
6 **Other Features**	*The game will continue until the plane collides with some barrier by instantiate and destroy the barriers*

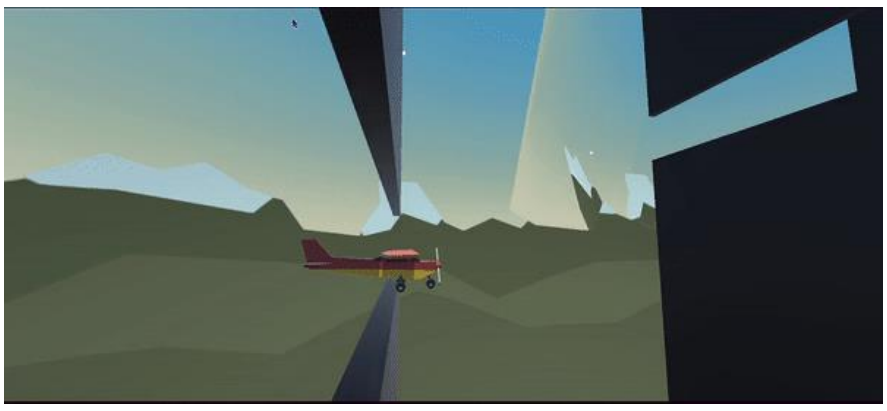
PROJECT TIMELINE

Milestone	Description	Due
#1	- *Getting the assets from Unity Asset Store*	*11/5*
#2	- *Equip the assets and make them as prefabs*	*11/8*
#3	- *Add rounding for fan and coins, and add cubes for scoring and Instantiation and destroying by script*	*11/12*

#4	- *Writing script for scoring, increasing and decreasing the speed, sound playing, instantiation and destroying, and game ending.*	*11/16*
#5	- *Making the UI*	*11/20*
Backlog	- *Adding sound when the plane takes the coin* - *Adding animation on the plane* - *Adding wind sound that increases by increasing the speed* - *Adding UI* - *Adding number of player life*	*11/22*

PROJECT SKETCH





Testing Schedule:

Test Case Type	Description	Test step	Expected Result	Status
Instantiation	Script	Play the game and skip the obstacles	When the plane skips some obstacle, the obstacles will instantiate and shown	Pass
Destroying	Script	Skip the last obstacle and going to the new obstacles	He must see the obstacles while it's destroying in the scene	Pass
Sound	Feature in unity as audio source	Play the game	He must hear the sound when he turns on the game	Pass
Coins	Assets in the scene that the plane can collect them	When the plane on collision with them	The coins must disappear when the plane collects them and every coin has effect	Pass
The movement of the plane	Using the vertical arrows, the plane can go up and down	Pressing the arrows	The plane must go up when he presses up arrow and it must go down when he presses it down	Pass
Back to the first position	In the scene, when the plane made collision with some obstacle, the game must return it to its first position	Making collision with the obstacles	The plane must be returned into its first position	Pass

PROJECT DESIGN DOCUMENT

24/1/2023

Mahmoud Rumaneh

PROJECT CONCEPT

1 **Player Control**	You control a	in this	
	Employee	*Isometric*	game
	were	makes the player	
	arrow keys (horizontal and down arrow) and space	*The horizontal arrow keys move the player right and left and the down arrow makes the player roll over, and the space key makes the player jump*	

2 **Basic Gameplay**	During the game,		from
	Obstacles and coins	appear	*Front of the player*
	and the goal of the game is to		
	The player collects the coins to make him rich but the obstacles might stop him to do this, each obstacle will lose the player's life when the player collides with it		

3 **Sound & Effects**	There will be sound effects	and particle effects
	-Upbeat music throughout the game as background *-Sound effect for collecting the coins will appear when the player collects the coin* *-Sound effect when the user's game over*	*-When the player collects 100 of coins the game will be faster by 2.* *-When the player collects 200 of coins he will win.*

4 **Gameplay Mechanics**	As the game progresses,	making it
	The player can jump by animation will turn on by the script and the rolling down also the same thing	*Making the player skip the obstacles*

5 **User Interface**	The	will	whenever
	-score *-player life*	*-increase* *-decrease*	*-when the player collects the coins* *-when the player collides with some obstacle*
	At the start of the game, the title		and the game will end when
	"Collect the coins"	will appear	*The player life equal zero or collects 200 of coins*

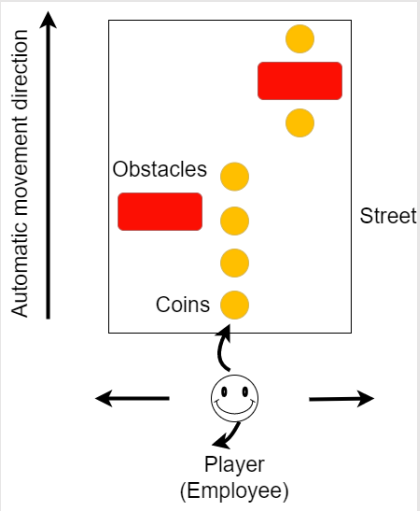
6 **Other Features**	*The player can't move out of the street, also there's dying animation when he loses the game*

PROJECT TIMELINE

Milestone	Description	Due
#1	- *Getting the assets from Unity Asset Store*	*13/1*
#2	- *Equip the assets and make them as prefabs*	*14/1*

#3	- *Add character and his animations, and add prefabs for all the assets for the instantiation and destroying by script*	*16/1*
#4	- *Writing script for scoring, increasing the speed, sound playing, instantiation and destroying, and game ending.*	*20/1*
#5	- *Making the UI and its scripts and scenes*	*25/1*
Backlog	- *Determining the appropriate mass for the player* - *Adding background for the UI* - *Adjusting the player's joints so that his movement is correct* - *Removing the rotation in the position of the animation*	*26/1*

PROJECT SKETCH



Test schedule:

Test Case Type	Description	Test step	Expected Result	Status
Instantiation	Script	Play the game and skip the obstacles	When the player skips some position at the first, all the road must instantiate and shown	Pass
Destroying	Script	Skip the last obstacle and going to the new obstacles	He must see the road while it's destroying in the scene	Pass
The road movement	Script	Turn on the game	All the road must go forward to the player	Pass
Animation	Animator working by the script	Turn on the game and press on the "Space" or "S"	The player must run at turning the game on and when he presses "Space" the player must jump and with pressing "S" he must roll.	Pass
Sound	Feature in unity as audio source	Play the game	He must hear the sound when he turns on the game, collects the coins, loses the game, and wins the game	Pass
Coins	Assets in the scene that the player can collect them	When the player on collision with them	The coins must disappear when the player collects them and it will be shown in the UI	Pass
The movement of the player	Using the horizontal arrows, the player can go right and lift	Pressing the arrows	The player must go right when he presses right arrow and he must go left when he presses it left arrow	Pass
Win the game	Script	Collecting 200 of coins	UI must be shown and with "You Win" text and sound with restart button	Pass
Lose the game	Script	Making collision with the obstacles in 3 times	UI must be shown and with "Game Over" text and sound with restart button	Pass
UI	Script and scenes	Wining or losing the game, or using the menu or credits	UI must be shown in these conditions with texts, colors, image, and buttons	Pass

PART 4

P6

Everything that I wrote in the project design document was applied in the playable demo. At the beginning of the game, a UI screen appears on which the name of the game is written, which is “Collect the Coins”, and a button written on it "Start" when you press it, the game starts and the player starts running automatically through the animation. And that he is running in his place and the street and all the other assets are moving towards him, which shows the player that he is running forward, and can collect the money. At the moment of collecting it, a sound appears and the amount of money he has collected appears as a score in the upper left corner of the screen. It also shows him directly below it how many lives it's remaining from 3 for him, which decreases by 1 when he collides with a barrier. The player is able to jump and roll to pass the obstacles, and he can win the game when he gets a score equal to 200.

Still, when he gets a score of 100, the speed of the game increases to double, so if he lost all the remaining player's lives, he will lose the game and a screen will appear to him that says Game Over with a voice of a person saying it and the appearance of animation for the death of the player. This is the game in short and everything I wrote in the Game Design Document has been applied in the standard way.

M5

I tried the game with my friends and family to try it and express their opinion on it and their reaction was different and their opinions were also different, some of them liked it very much and they liked the music, the way to run, jump, rolling, the sound of collecting money and how the player loses life, some of them said that the movement of the player to the right and left is not beautiful, it must there should be an animation that shows the player’s movement to the right and left, not just the translation of the player’s location, and my friend told me that the UI needs more details and greater aesthetics by adopting a fixed theme for the whole game with fixed colors, and all the assets inside the game must contain a fixed theme and uniform colors, then the game needs development in terms of design and that the physics is sometimes not perfect and needs some improvements in the jump and rolling.

Also, someone told me that I have to add levels in the game and add a suspense to it and change the scene as a theme after the player passes a certain time, they told me a lot of constructive and positive comments. They liked the details of the green grass, the mailbox, the telephone, the trees, and these details, but the game needs development.

M7

My evaluation of the game’s output was based on the user’s expectations through the game design document, as he was expecting a beautiful game as I explained its idea and that it is easy to play and its rules are easy and can be completed on among friends, the Collect The Coins game aims to entertain and empty the player’s time with something entertaining and gives a feeling of happiness, even if it was a simple feeling, this is my goal and the goal of the game, but definitely it must be developed and its strengths and weaknesses must be identified. Its current situation must be determined in order to understand what we must modify and develop and what we must strengthen. One of its strengths is the ease of play, as any age group can play and enjoy it. It also contains clear and eye-friendly graphics, which makes the player’s experience comfortable, and one of its strengths is also the presence of some details in the street to make it more realistic through green grass, mailboxes, phone boxes, trees, and so on, and the barriers that the player crosses are realistic and can be in any real street, and the character of the player in the game is also realistic, as he is a person wearing a formal suit, and the animations that are used in the game represent a realistic and

natural movement of the human body, and to be more realistic I added sounds such as a background sound, a sound for coins, for winning and losing, and so on, and UI screens to show the player how to play and navigate to them, these are the strengths that I see in Collect The Coins game.

As for the weaknesses, I see that the Collect The Coins game has weaknesses in the physics of the movement that the user makes, as it is not 100% accurate, but I tried hard to match reality by increasing the mass of the player so that he would not be affected by the “Rigidbody” and the transitions between each animation so that the game continues smoothly, I encountered many problems and solved them to reach the best version that I can offer so far, I also see that the game has weaknesses in the UI because all screens contain the same image, which I do not see as the best image that can be placed for the game, but a special image must be designed for this game, as well as the font type, font size, button shape, and color, needs more details and the addition of many screens that make the player's experience more entertaining, in addition for this the animation in the rolling, it's not clear that this the animation for rolling.

D3

To identify the strengths and weaknesses of a particular game, this is a very useful thing for developing it and knowing its current location, and the player must also intervene in determining these strengths and weaknesses, because whoever tries the game as a designer or developer will not know all the mistakes, so it is not a requirement that you be a good designer and developer and a good player also. But regarding the “Collect The Coins” game, I identified its strengths and weaknesses through my experience with it and the opinion of the people who tried to play it. Certainly, there were weaknesses that the player did not expect, and there were strengths that he did not expect as well, and all of this is heading towards one goal. It is the development of the game and show it better, the development of the game and keeping pace with development is the thing that determines its success.

Failure to identify these strengths and weaknesses of any game detracts greatly from the player's experience and fails to meet their expectations. For example, the game must not be too easy or too difficult for the player to find a suitable environment for competition and challenges, and the game must have clear storytelling and a specific goal. It is completed at the end, and the sound and video quality have a great impact on improving the player’s experience, increasing his satisfaction with the game, making his experience long-term, and telling his friends about it. The player’s opinion must be taken seriously, understanding the different opinions of the players, and reaching a solution and development that satisfies the majority and achieves great success.

The developments that I will make to the "Collect The Coins" game will greatly change it for the better, by starting to develop the game itself by improving the sound quality and making the game have a greater goal by telling a complete story to it and adding different challenges and stages, which increases the player's fun and satisfaction, also, improving the animation and adding more realistic animations, especially in the lateral movement of the player, and adding assets such as street lighting, the shape of a real sky, and additional street components.

Also, the physics in the game must be developed and the movement of the player better, after that many improvements must be added to the UI such as changing images and adding new screens that appear at certain stages of the game and bear a game-specific design and not just pictures, the UI as a whole needs many improvements and designs and adding elements that help the player understand the game more, such as a screen that shows what buttons the game uses and what each button does.

REFERENCES

[1] Schardon, L. (2023) What is unity? – A guide for one of the top game engines, GameDev Academy. Available at: https://gamedevacademy.org/what-is-unity/ (Accessed: January 12, 2023).

[2] Steiger, P.by A. and Published by David LayzelleWriter - Tech Guru (2020) Evolution of unity and how it became a great tool also for artists and designers, Unity Developers. Available at: https://unitydevelopers.co.uk/evolution-of-unity-and-how-it-became-a-great-tool-also-for-artists-and-designers/ (Accessed: January 13, 2023).

[3] What is Unreal Engine?: How it works: Scope & career: Advantages (2022) EDUCBA. Available at: https://www.educba.com/what-is-unreal-engine/ (Accessed: January 14, 2023).

[4] Usaid (2022) How unreal engine 5 can impact the games industry, GamingBolt. GamingBolt. Available at: https://gamingbolt.com/how-unreal-engine-5-can-impact-the-games-industry (Accessed: January 14, 2023).

[5] Buckley, D. (2023) Unity vs. Unreal – choosing a game engine, GameDev Academy. Available at: https://gamedevacademy.org/unity-vs-unreal/ (Accessed: January 15, 2023).

[6] Game engines and game history (no date) Kinephanos. Available at: https://www.kinephanos.ca/2014/game-engines-and-game-history/ (Accessed: January 16, 2023).

[7] Game engines (types & purposes) (2013) Jahmel Coleman. Available at: https://jahmelcoleman.wordpress.com/games-development/game-engines/ (Accessed: January 16, 2023).

[8] Danley, K. (2022) Future role of game engines in gaming industry, Straits Research. Straits Research. Available at: https://straitsresearch.com/article/game-engines-the-new-future-trends-in-gaming-industry (Accessed: January 17, 2023).

[9] Castello, J. (2022) Fortnite is turning into a digital theme park, Polygon. Polygon. Available at: https://www.polygon.com/23497347/the-future-of-fortnite (Accessed: January 17, 2023).

[10] Design, O.W. (2019) Fortnite facts for game designers, Design Tutorials and Articles. Available at: https://www.onlinedesignteacher.com/2019/01/fortnite-facts-game-designer-should-know.html (Accessed: January 18, 2023).

[11] D'Anastasio, C. (2019) Report: Fortnite developers describe severe ongoing crunch, Kotaku. Kotaku. Available at: https://kotaku.com/report-fortnite-developers-are-severely-overworked-1834243520 (Accessed: January 18, 2023).

[12] Lovering, N. (2022) Is fortnite bad for kids? effects of violent video games, Psych Central. Psych Central. Available at: https://psychcentral.com/lib/more-evidence-fortnite-is-bad-for-your-childs-health (Accessed: January 18, 2023).

[13] Ltd., A. (no date) What is a gaming engine?, Arm. Available at: https://www.arm.com/glossary/gaming-engines (Accessed: January 19, 2023).

[14] Evaluation of game engines for simulated clinical training (no date). Available at: https://www.researchgate.net/publication/228368856_Evaluation_of_Game_Engines_for_Simulated_Clinical_Training (Accessed: January 19, 2023).

[15] Schardon, L. (2023) What is unity? – A guide for one of the top game engines, GameDev Academy. Available at: https://gamedevacademy.org/what-is-unity/#Key_Features (Accessed: January 20, 2023).

[16] Features (no date) Unreal Engine. Available at: https://www.unrealengine.com/en-US/features (Accessed: January 21, 2023).

[17] Temple run (2023) Wikipedia. Wikimedia Foundation. Available at: https://en.wikipedia.org/wiki/Temple_Run (Accessed: January 22, 2023).

[18] Technologies, U. (no date) Hitman go, Unity. Available at: https://unity.com/case-study/hitman-go (Accessed: January 23, 2023).

[19] Angry birds (2023) Wikipedia. Wikimedia Foundation. Available at: https://en.wikipedia.org/wiki/Angry_Birds (Accessed: January 24, 2023).