VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

3D FIGHTING GAME WITH UNITY

By

Mai Lê Hùng – ITITIU19125

Nguyễn Hoàng Đạt – ITITIU19097

Lê Tấn Đạt – ITITIU18270

A thesis submitted to the School of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Bachelor of Information Technology/Computer Science/Computer Engineering

Ho Chi Minh City, Vietnam
Year

3D FIGHTING GAME WITH UNITY

APPROVED BY:

_____
(*Typed Committee name here*)

_____
(*Typed Committee name here*)

_____
(*Typed Committee name here*)

_____
(*Typed Committee name here*)

THESIS COMMITTEE
(Whichever applies)

ACKNOWLEGMENTS

I thank Dr. Nguyen Van Sinh's expert assistance with the utmost respect and admiration. He gave me ongoing motivation and assistance, which enabled me to succeed.

My appreciation is extended to Dr. Le Duy Tan, the laboratory's director. These were excellent years for studying thanks to his technical assistance and sense of humor. I am appreciative to the staff at the International University's School of Computer Science, especially Dr. Nguyen Van Sinh, who has helped me since I first started my studies. I would also like to thank the people on my reading and testing committee.

Note: Paper A4, Top: 2.5cm; Bottom: 2cm; Left: 3cm; Right: 2cm

Table of Contents

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1. Background

With the formation and development of today's game. We can build and shape games easily. There are different types of games.

The topic of game fights is one of the most popular genres today. is a type of game that interacts with the physical strength between the animations built into the game. And for the game fight, this time is a 3D simulation game between a player interacting with bots that form in the game to physically interact with each other. Players will form in the form of a fully materialized character and fight bots.

The scale of the image construction of this project is relatively 3D and with minimal processing pulses but can bring a sense of excitement that can help players freely interact with characters, everything built in this game fight.

1.2. Problem Statement

To effectively build this fighting game, we need to research and analyze the essential interfaces, important goals in the development of each part of the project, problems that need to be mentioned such as:

+ Interactive frame between players for the game on how to best balance interaction. usually have window mode or full-screen mode so it can be easily selected. It also distributes pixel profiles according to the normative ratio so that interactors can choose.

+ Select and distribute the game's 3D animations to bring out the right vision and game scene fit.

+ Set up 3D character images to build as meticulously and objectively as possible when projected onto the frame for players to see clearly.

+ Bring the player's attraction to the game with an interactive simulation of the character's function, movement, and functional actions in the fight game.

+ Can optionally set up the wrong modes to improve the new experience, and continue to update the wrong sessions to improve the power of the game.

1.3. Scope and Objectives

For this fighting game project, the specific goal is that in the project we can create specific 3D characters and choose harmonious color images. Bringing interaction between players by creating in-game characters and physically interacting with the machine bot is formed. The goal is to destroy the bots and complete the character upgrade to the next level. Mainly create entertainment for players and do not place heavy emphasis on violence against humans. In addition, the main goal is to apply the knowledge gained from the Computer Graphics course in this project

1.4. Assumption and Solution

In short, this fight game project is released with the goal can create a 3D character. Players choose the animation they like and interact with the physical operations of the bots created in the game, completing the levels will upgrade the character to a new level. To accomplish the goals set out above, we need to develop 3D images of the characters, which can be modeled after street fighter characters. game, create a multi-pixel frame layout to bring the best experience to the player. After building 3D frames of characters and bots in the game. Proceed to put frames and implement animations on Unity 3D, combining animations into the same project. Then to link the animations to a specific game, the actions are functionally implemented via C# and also linked to the characters to run and create the complete fight game project.

1.5. Structure of thesis

The project layouts are divided into the following sections:

+ The work related to the description of the game industry and 3d graphics will be formed in chapter 2.
+ Chapter 3 will describe the design methods and graphics algorithms that need to be described to form this game project.
+ Implementing the descriptions, implementing the designs according to Chapter 3, and producing real results in Chapter 4.
+ Chapter 5 is a re-test of the entire component of the project from the visuals and interactions between the game and the player. At the same time, discuss and evaluate the success of the project.
+ Chapter 6 Summarize the whole process of building this fight game project. Give an overall review between game interaction and players. Development orientation continues for a long time with new updated versions of the game.

CHAPTER 2

LITURATURE REVIEW/RELATED WORK

2.1. The growth of gaming insdustry

Currently, the online game industry has made new strides. Especially e-sports are a competition organized entirely from the socialization budget. E-sports has a total of famous games such as: League of Legends, Lien Quan Mobile, FIFA Online 4, Free Fire, LoL Wild Rift, PUBG Mobile, Mobile Legend Bang bang and Raid.

In addition to the number of online games released by the world constantly increasing, the revenue of businesses providing online games online is also increasing. The most obvious thing is that besides creating tens of thousands of jobs, the game industry is also creating a creative industry, exploiting digital content from games.

However, at present, the management of online games in fact still has many holes such as age restrictions, game time, unauthorized games leading to many unfortunate events.

In the increasingly volatile market situation, the Government has come up with many new solutions for this market. We need to analyze game policy and the environment. At the same time, it is necessary to study and develop a method of management and approval of online games, including mobile games, to standardize the order of publishing and operating online games. Develop regulations on online game catalogs to be released in the coming year. This regulation aims to control and grasp the situation of online game publishing in advance so that ministries and branches have an overview, in-depth as well as forecast its impacts. promulgate policies that contribute to stabilizing and regulating the online video game market to operate effectively.

Finally, experts say that authorities need to improve management, review, and guide businesses to standardize their activities. At the same time, support the development of cultural industries, especially the online game industry. Having policies to encourage and support non-public cultural enterprises to participate in online game publishing, and to encourage and support online game publishers. Strengthen forms of cooperation with media publishing companies.

2.2. Graphic Desgin

Graphic design is a form in which designers are free to choose, create, and arrange visual elements such as illustrations, photographs, writing, and lines on a surface to create and communicate. load a message.

Graphic design in general is widely used in the world of advertising, packaging, feature films, and more and is divided into categories:

+ Publication design: including book cover design, magazine cover design, poster design, flyer, brochure, advertisement, business card, and other publications.

+ Website Design

+ Movies include CDs, DVDs, clips…

+ Brand design: logo and brand identity…

+ Product packaging design

2.2.1. The model 3D of graphic

Three-dimensional (3D) models use a collection of points in 3D space connected by various geometric objects, such as triangles, lines, and curved surfaces, to represent a physical body. A 3D model can be produced manually, algorithmically, or by scanning a set of data. Texture mapping can be used to further characterize their surfaces.

3D graphics and CAD both extensively employ 3D modeling. They were used before 3D graphics on home computers became widely used. Before computers could create 3D models in real-time, many computer games used pre-rendered representations of those objects as textures. The designer can then view the model from a variety of angles and perspectives, which can aid in determining whether the object was made according to the designer's original vision. This way of looking at the design can aid the designer in identifying any changes or enhancements that the product needs.

In many fields today, 3D models are used. The health industry uses intricate organ models, which can be produced using numerous 2D slices from an MRI or CT scan. They are used in colorful and realistic motion pictures as characters and props. They serve as resources for computers and video games in the video game business. They serve as extremely accurate models of chemical substances in science. Instead of using traditional physical architectural models, architecture employs them to show potential buildings and landscapes. They are used by the engineering community for many different purposes, including the creation of new machines, vehicles, and structures. In recent years, 3D geological models have become the norm in the earth research community. Additionally, physical devices created with CNC machines or 3D printers can be based on 3D models.

2.2.2. Animation

Animation is a method of photographing a series of paintings, models, or even puppets to create the illusion of movement in sequence. Because our eyes can only hold an image approximately. In 1/10 of a second, when several images appear one after the other, the brain combines them into a moving image. In traditional animation, images are drawn or drawn on transparent celluloid sheets for photography. Early cartoons are examples of this, but most animated films today are created using computer graphics or CGI.

Advantages of 3D animation:

+ Realistic, vivid images
+ Creative and unique ideas
+ Save time, reasonable price

Types of Animation:

+ Traditional Animation
+ 2D Animation

+ 3D Animation
+ Motion Graphics
+ Stop-Motion

CHAPTER 3

METHODOLOGY

3.1. Overview

In this project, we will use Unity and C# to do everything in this game. First about Unity, we will use it to background everything about graphics, sound, animation, ... in the game. As for the C# language, we will use it to regulate and manage all the mechanics of the game including movement, attacks, combos, and AI for enemies, ... Because just getting used to it. We have been working with Unity and C# since receiving this project, so our knowledge of these programming languages is extremely limited so all methods will only be applied at a basic level. The knowledge applied from the Computer Graphic subject will also be applied including shapes in 3D space, displacements, rotations, ... shapes, and colors in computer science, ... All the knowledge learned will also be applied in this project

3.2. User requirement analysis

After sitting together and analyzing, we have agreed on user requirements for current games, encapsulated in the following elements: Graphics, Animation and motion, Sound, Combat/Combo mechanics.

3.2.1. Graphics

About the graphics in this game, we will use completely 3D graphics so the focus will be on the authenticity of both the surface and the depth in the generated cubes. With the background as an urban area, focus should be on the truth in space with the shapes of buildings, trees, garbage, ... created with an emphasis on depth in space to create a sense of completeness. completely different from 2D graphics. There is also a harmonious color scheme that accurately represents the dystopia context that the game brings. Therefore, while creating the human lips in the game, we pay great attention to the depth of the scene's spatial representation, which gives the player a real sense of 3D space.

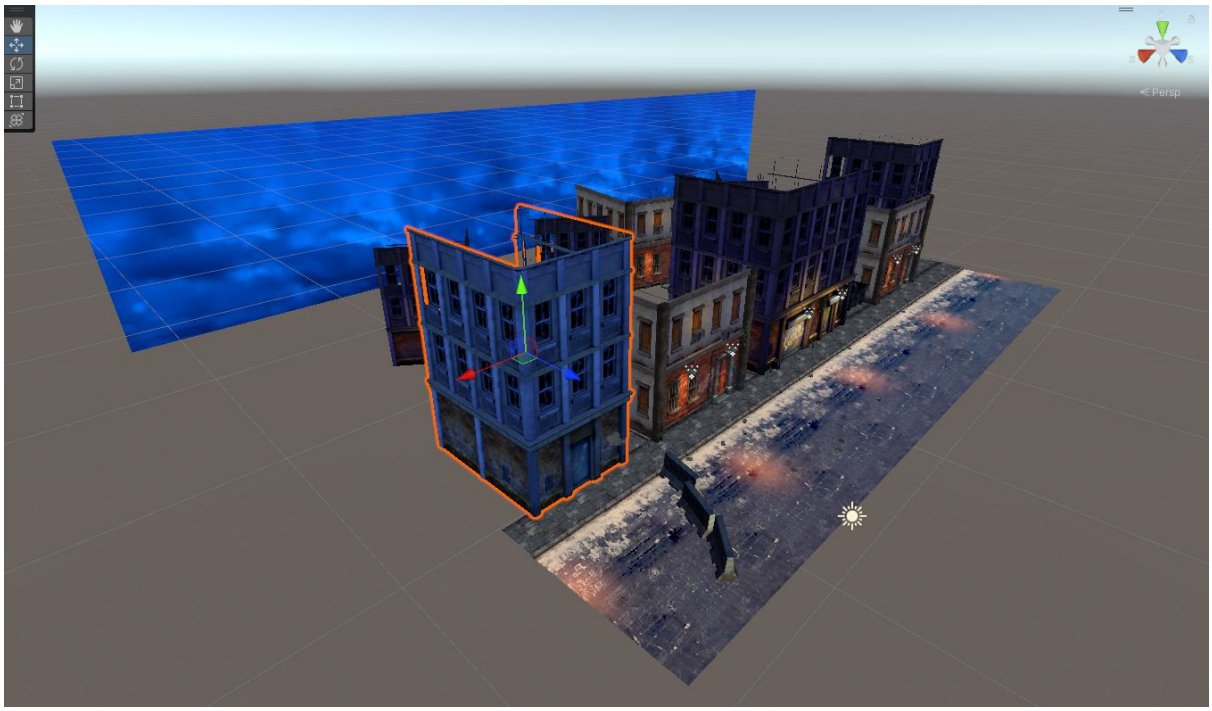

Figure 1 Enviroment of Game at Right Side

Figure 2 Enviroment of Game at Center Side

As for character creation, we try to create a character with a strong appearance, muscles, and body curves that bring a feeling of both strength and authenticity to the player. We then colorize the characters to bring out the dusty feel of the street as much as possible

Figure 3 Create Player model and at Mat-cap skin

3.2.2. Animation and motion

Next are the game's animations. The current user's need is the smooth movements of the character, in addition, these movements must be strong to show the ability as well as the person of the character in the game. That's why we researched a lot of martial arts punching and kicking poses and created amazing animations. In addition, to increase the realism and smoothness of the character, we also meticulously added every movement of every part of the character's body on each frame, even as small as our fingers. let it move frame by frame. It is these things that create extremely smooth and powerful animations in this project

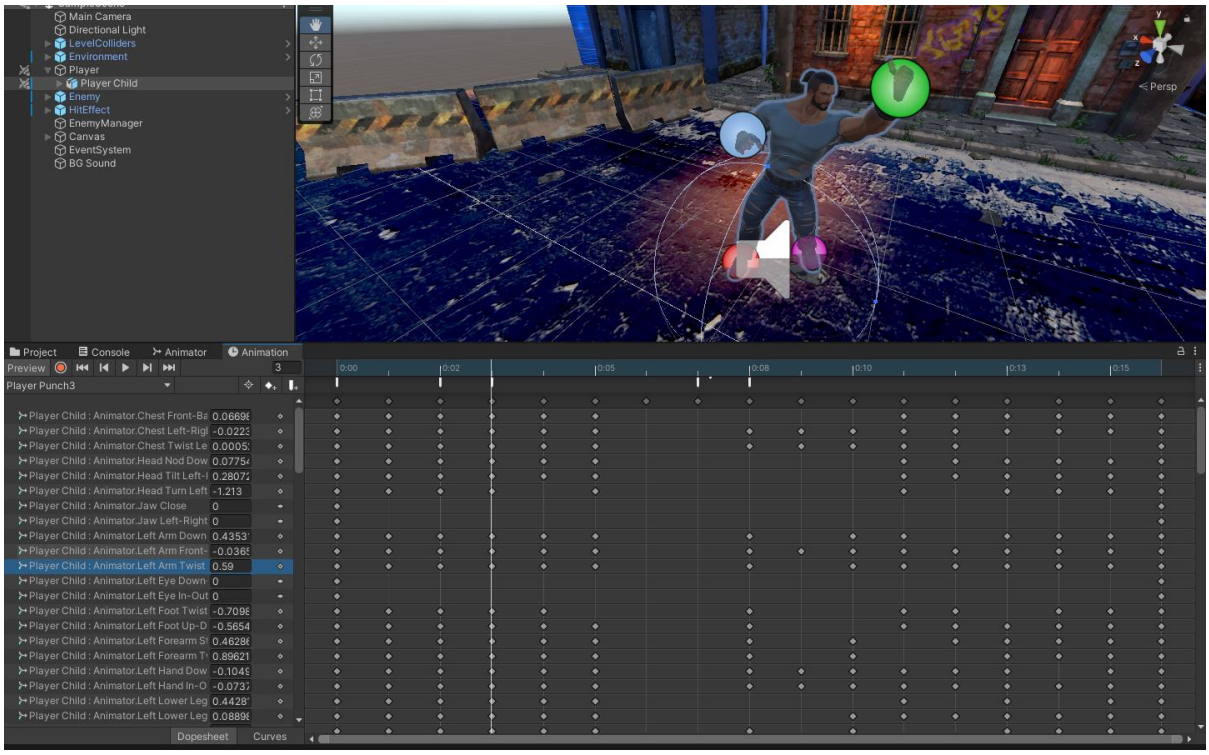

Figure 4 Animation Player Punch 3

3.2.3. Sound and sound effect

Sound is also an extremely important factor that today's users are also extremely interested in. Imagine a fighting game without any sound, right? Therefore, to increase the drama of the game, we have added an extremely heroic and tense background sound to the game. There are also sound effects when the player punches or kicks to increase the excitement and action when the player fights. Moreover, sound effects are also added when the player knocks down or kills the enemy, making the battle extremely realistic and attractive.

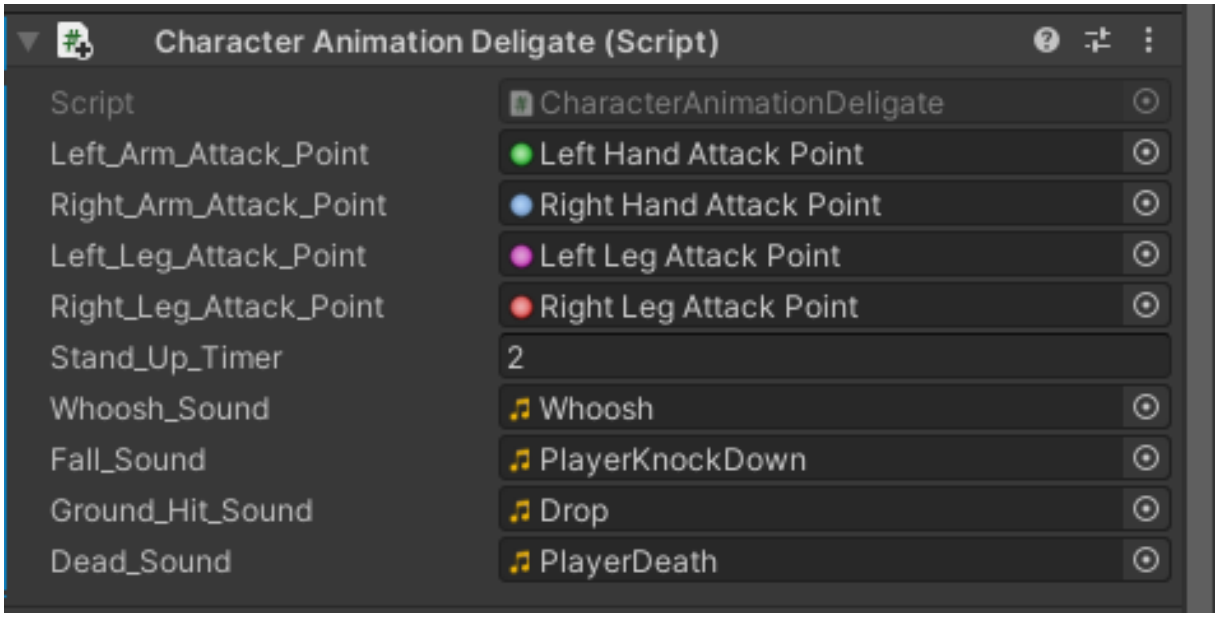

Figure 5 Character Animation Delegate (script) control sound files

3.2.4. Combat/Combo

Finally, the battle mechanism of the game. If a game only had punches and kicks, players would be bored, wouldn't they? That's why we've added combo mechanics to our game. When the player presses the exact order of the buttons, a sequence of combos is extremely beautiful. Also, we know that you guys really like random mechanics so we've added random mechanics to our combos. Specifically, when hitting the opponent with a combo sequence, there is a possibility that the opponent will be interrupted (hitted) or even knocked down for a period.

3.3. System Design

3.3.1. Game Design Diagrams

This is the Class Diagram of this game:

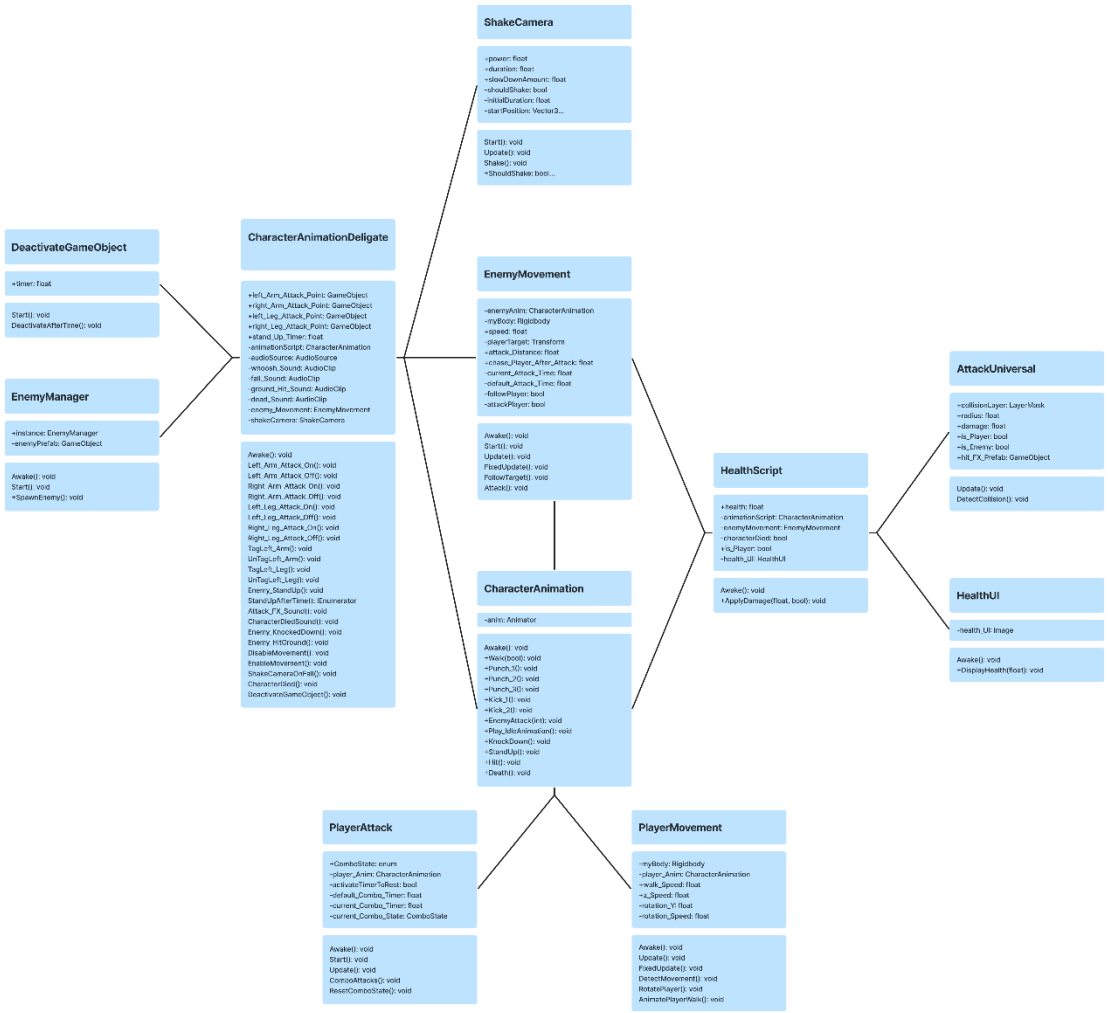

Figure 6 Class Diagram

(link to Classdiagram: https://www.figma.com/file/TYwlWBzaxNH8HMm4RADVSU/Untitled?node-id=0%3A1&t=T10QiF1cHsr0IxDA-1)

This is the Player Animations Diagram:

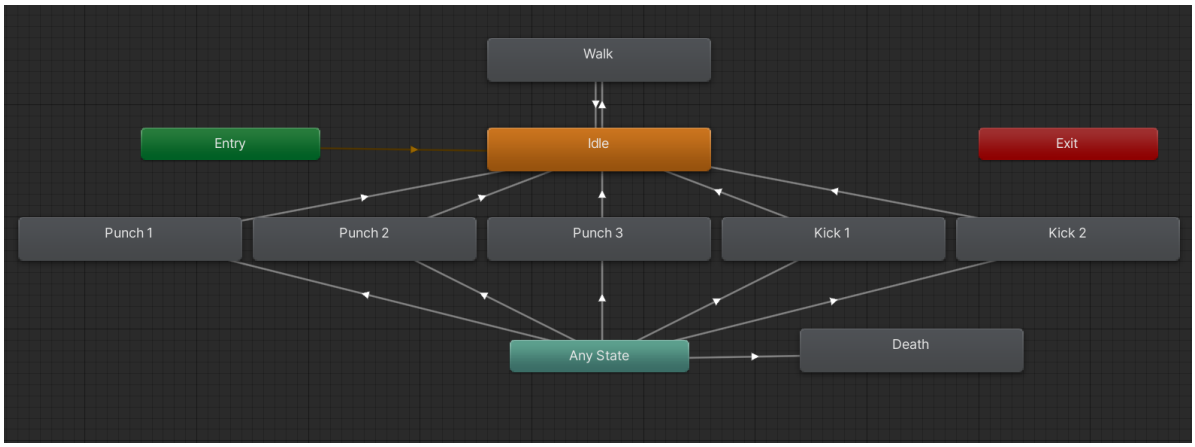

Figure 7 Player Animations Diagram

This is Enemy Animation Diagram:

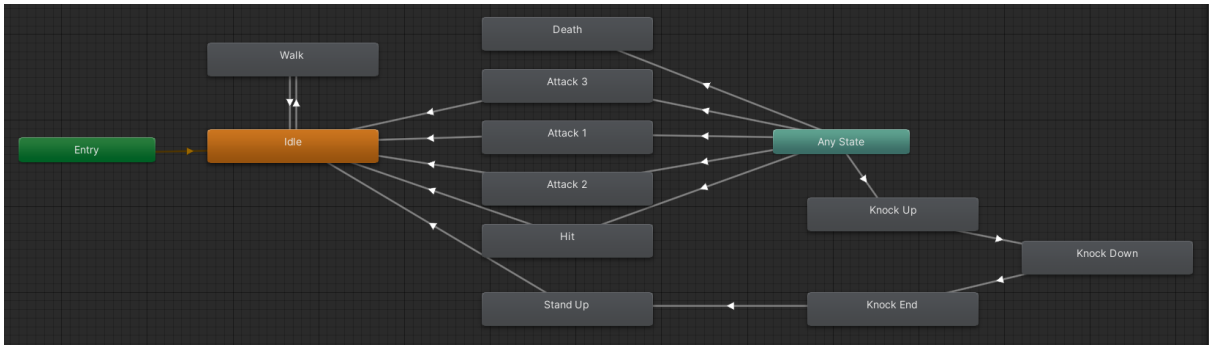

Figure 8 Enemy Animation Diagram

3.3.2. User Interface design

Finally, is the user interface. We took the element of convenience and recognizability. So, the layout so that the content can stand out from the background:

Figure 9 In Game Screen

1. Player Icon
2. Player Health Bar
3. Player Character
4. Enemy

CHAPTER 4

IMPLEMENT AND RESULTS

4.1. Implement

Now I will start explaining step by step how to make this game from inception to completion so that everyone understands.

4.1.1. Making Environment

First let's talk about creating environments in the game. I used Mat-cap to create shapes and color the entire environment in the game. To create a realism in the space I created a lot of different shapes of buildings, trees, trash cans, graffiti, etc. Finally, I created a sky background and left it. at the end to create depth in space:

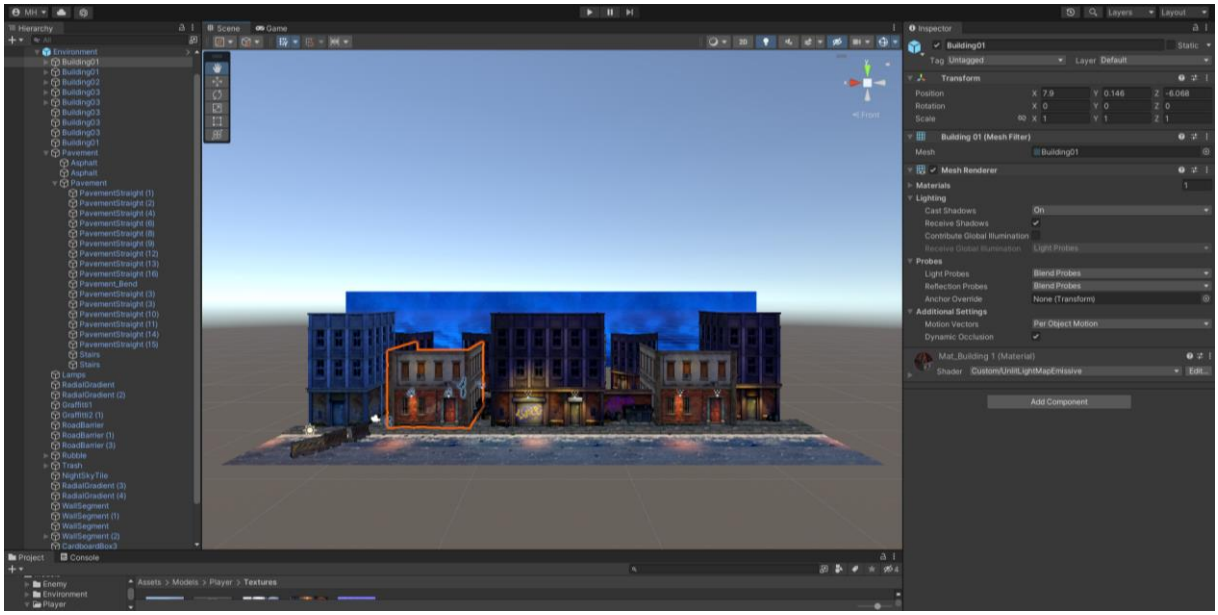

Figure 10 Making Environment of Game in Center Side

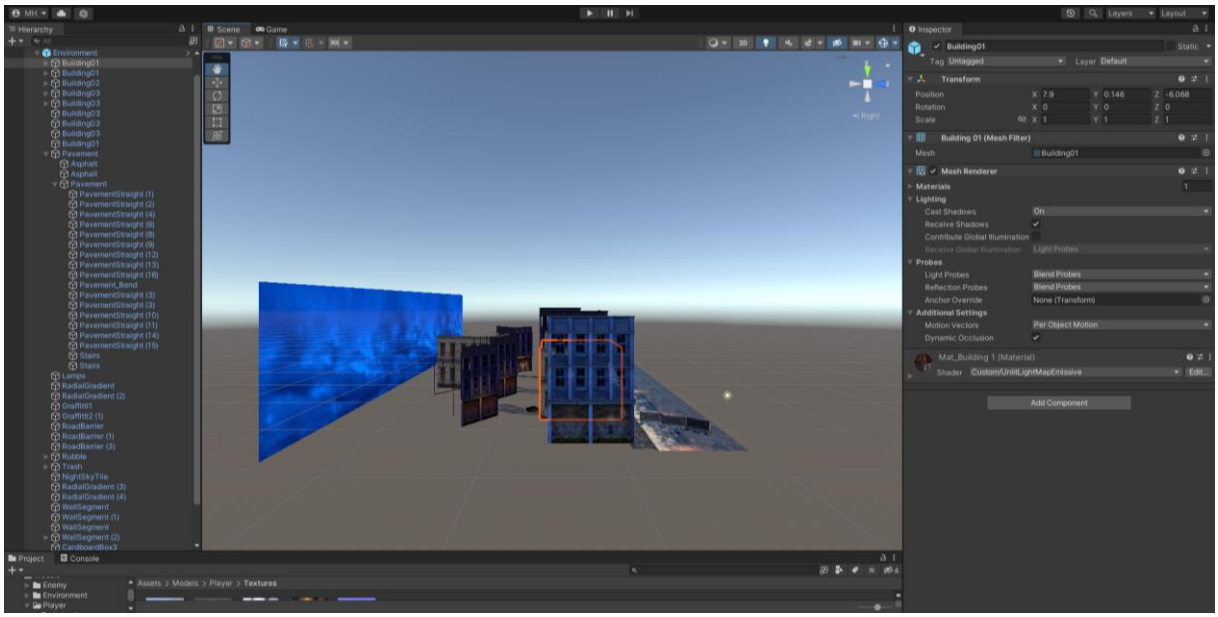

Figure 11 Making Environment of Game in Right Side

4.1.2. Create Player and Enemy Models

Next, we started shaping the player and enemy characters. To create realism for the character, we have divided the character into many small parts to each finger to shape the character. The division of parts makes the character able to create softness and truth in each part of the body to the whole.

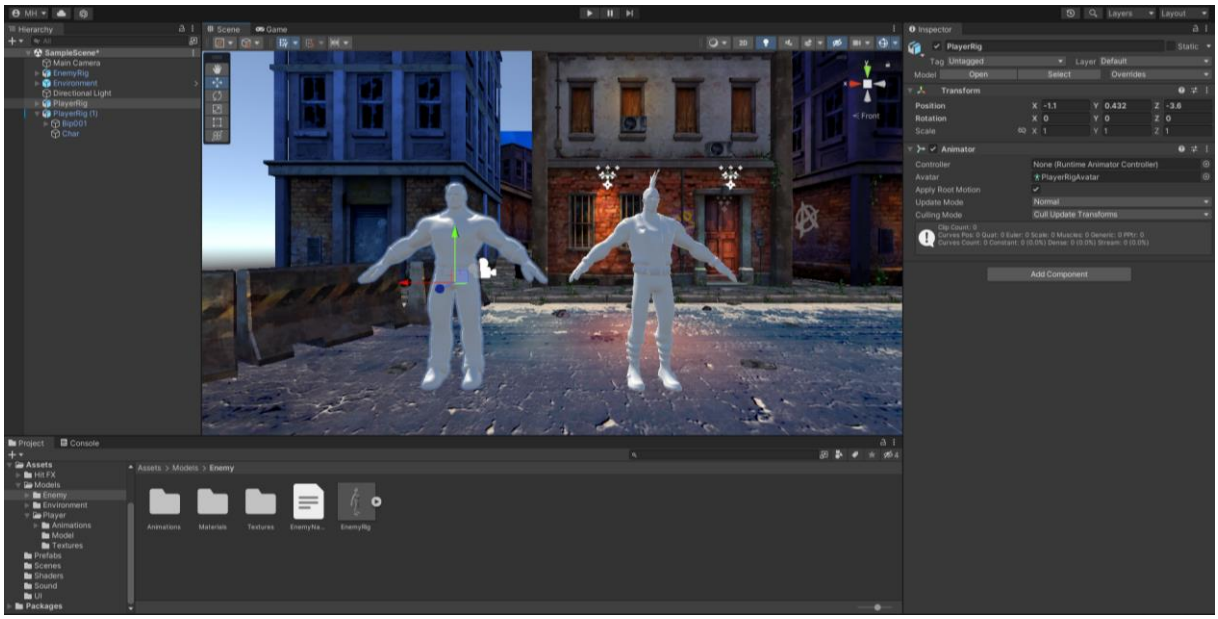

Figure 12 Creating Models

We then used Mat-cap to add color, shadow and light to the character to make it as beautiful as possible:

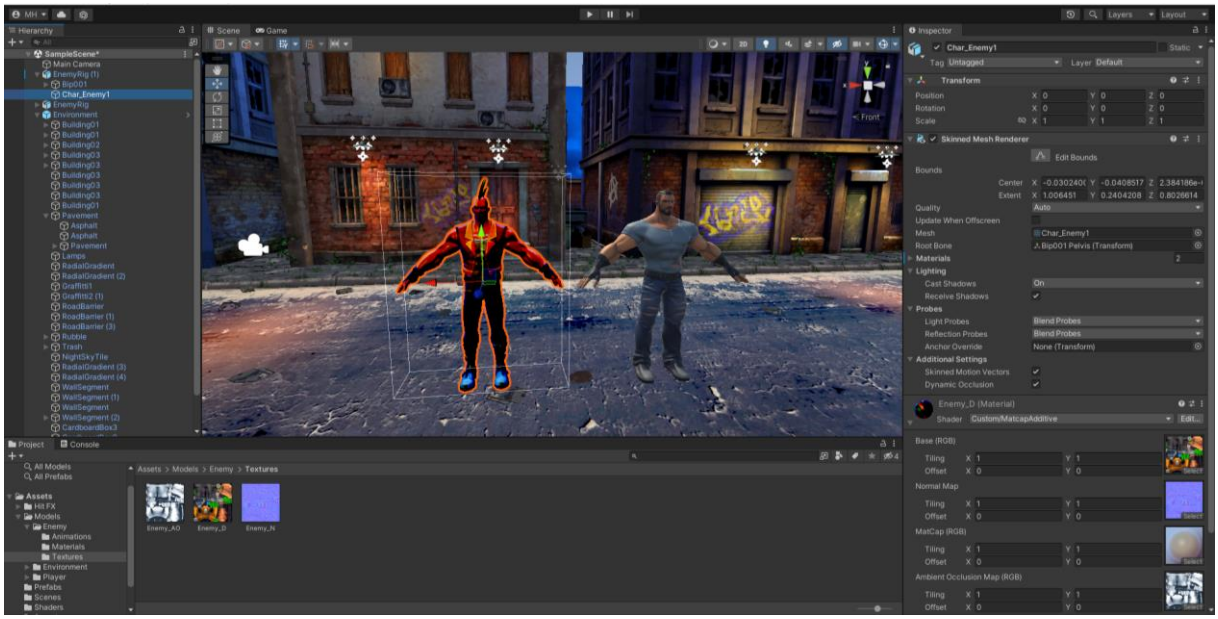

Figure 13 Adding Mat-cap Skin

4.1.3. Movement and Animation

About the movement we have created a class that is PlayerMovement to control the character through the buttons w, a, s, d. Also in that class we also add a method called RotatePlayer to rotate the character when the character moves up and down:

```
void DetectMovement(){
    myBody.velocity = new Vector3(
        Input.GetAxisRaw(Axis.HORIZONTAL_AXIS) * (-walk_Speed),
        myBody.velocity.y,
        Input.GetAxisRaw(Axis.VERTICAL_AXIS) * (-z_Speed));

}

void RotatePlayer(){
    if(Input.GetAxisRaw(Axis.HORIZONTAL_AXIS) >0){
        transform.rotation = Quaternion.Euler(0f, rotation_Y, 0f);
    }

    else if(Input.GetAxisRaw(Axis.HORIZONTAL_AXIS)<0){
        transform.rotation = Quaternion.Euler(0f, Mathf.Abs(rotation_Y), 0f);
    }
}

void AnimatePlayerWalk(){
    if (Input.GetAxisRaw(Axis.HORIZONTAL_AXIS) != 0 || Input.GetAxisRaw(Axis.VERTICAL_AXIS) != 0){
        player_Anim.Walk(true);
    }
    else{
        player_Anim.Walk(false);
    }
}
```

Figure 14 Player Movement Controll Code

Next, we created the character animations in as much detail as possible with changes in the smallest parts of the character like fingers from frame to frame. We then imported all the Player animations into the animator and started working on connecting the transition relationships between the character's moves and also adding timeout when changing moves. of character:

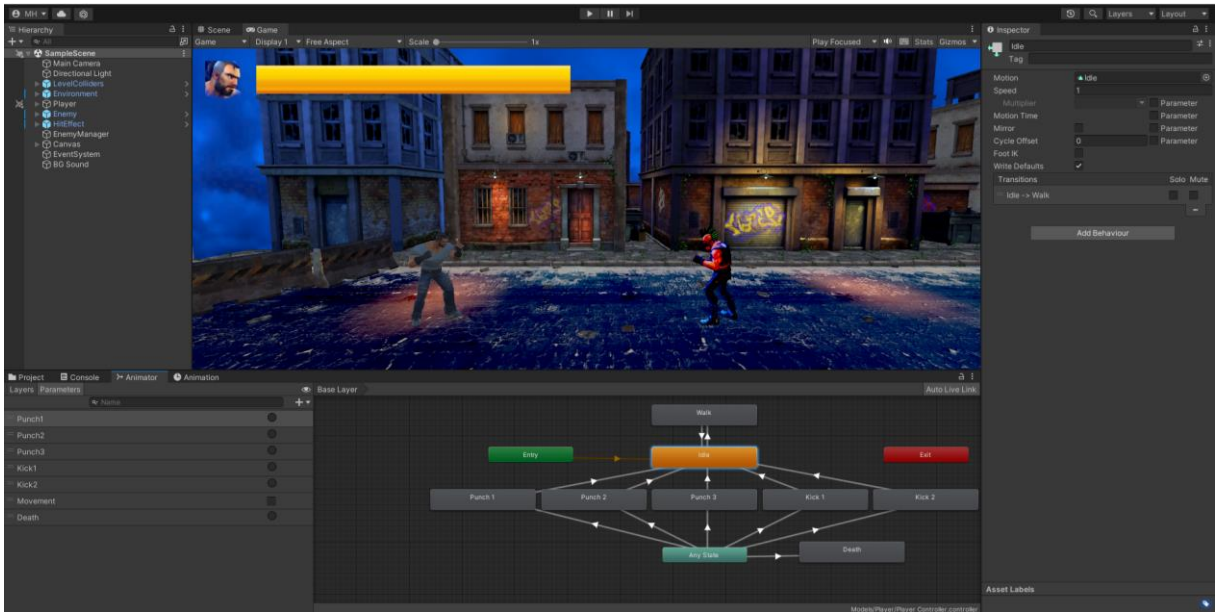

Figure 15 Player Animator

Regarding the enemy's movement because no one controls it, we created a class called EnemyMovement that is responsible for controlling Enemy to chase and attack the player. And the FollowTarget() method is responsible for this. This method will check that if the Player is within Enemy's attack range, Enemy will stop moving and attack the player. If the Player is out of Enemy's attack range, Enemy will move towards the Player's direction:

```
void FollowTarget(){
    if(!followPlayer){
        return;
    }

    if(Vector3.Distance(transform.position, playerTarget.position)>attack_Distance){
        transform.LookAt(playerTarget);
        myBody.velocity = transform.forward * speed;

        if(myBody.velocity.sqrMagnitude !=0){
            enemyAnim.Walk(true);
        }
    }
    else if(Vector3.Distance(transform.position, playerTarget.position)<= attack_Distance){
        myBody.velocity = Vector3.zero;
        enemyAnim.Walk(false);

        followPlayer = false;
        attackPlayer = true;
    }
}
```

Figure 16 Method FollowTarget() of Enemy Code

We then created the movement and attack animations for Enemy. Next, we import these animations into the Enemy animter and start establishing the same relationships as we did with the Player:

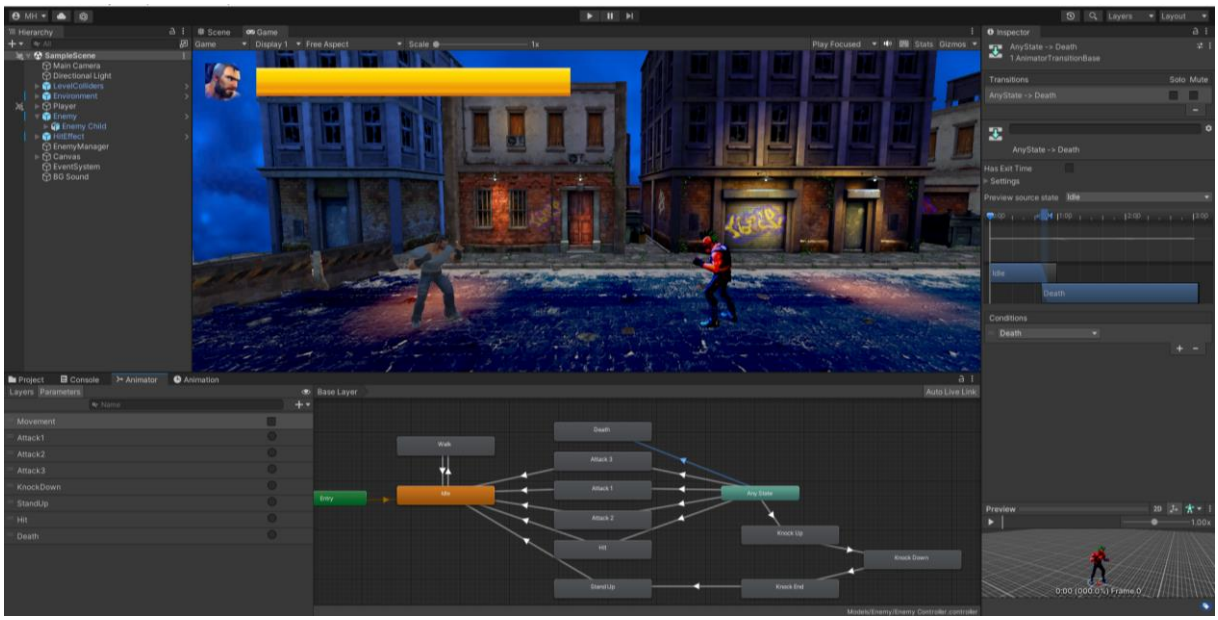

Figure 17 Enemy Animator

4.1.4. Attack/Combo and Hit box

First, we create the PlayerAttack class to control the Player's attacks. After a few basic settings, we have made the J button for punching, the K button for kicking. We then set up a method called ComboAttack() to create a continuous sequence of combos when the player presses the correct key combination. In it, there will be a timer, if the player presses the correct key combination and continuously within 0.4s, the combo will be performed, but if it is slower than 0.4s, it will only show normal attacks.

We then create a CharacterAnimation class to control all the states and animations of both Player and Enemy. You can simply understand it as methods that will call out the animations and characters that need to be shown.

Next, we create a class EnemyMovement to control both the movement and attack of Enemy. Regarding Enemy's movement we mentioned above, and for the attack part, we set Enemy to attack the Player if in range with a 2s attack speed and the attack will be randomly selected for 1 in 2 seconds. 3 attack animations that we created for the enemy.

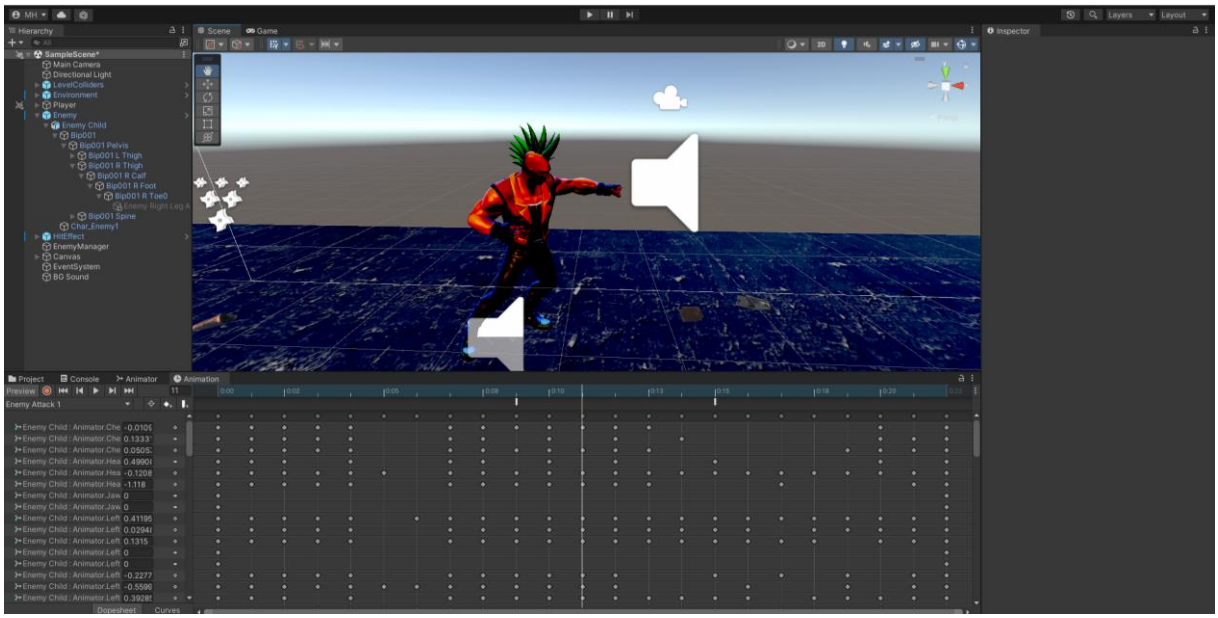

Figure 18 Enenmy Attack 1 Animation

Then we created HitBox for both Player and Enemy in 2 arms and 2 legs. Next we create the AttackUniversal class to manage the interactions between Player and Enemy objects. In this class, there are 2 attributes is_Player and is_Enemy to distinguish objects. At the same time, this class also has the effect of declaring the damage of each attack and the radius of impact of the attacks. In addition, it is necessary to create 2 more Tags for 2 objects to separate both objects, Player and Enemy. And by creating the above 2 tags also created 2 tags for the Collision Layer in the hit box of Enemy and Player. In the Player's hit box, select the Collision Layer as Enemy so that the hit box will only activate when touching the object with the Tag Enemy and do the same for Enemy.

Figure 19 Player Hit Box

However, that much is not enough because if you stop there, when the 2 characters move close enough, they will automatically damage each other without having to use attacks. That's why we created the CharacterAnimationDeligate class to handle this. In this class will create events in an animation of the kernel to mark the time when the hit box is activated and deals damage to the target hit. These events are added throughout the attack animations as follows:

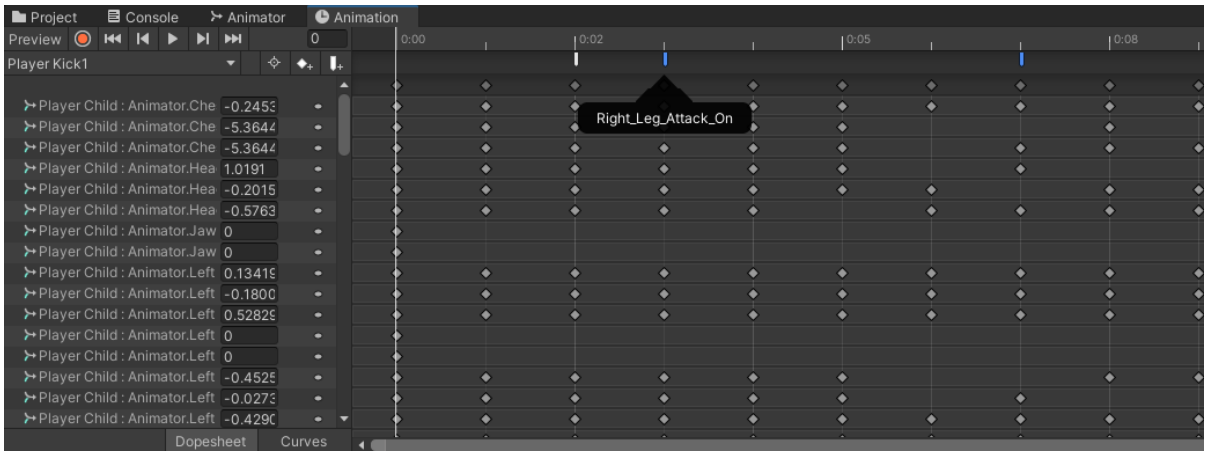

Figure 20 Right_Leg_Attack_On event

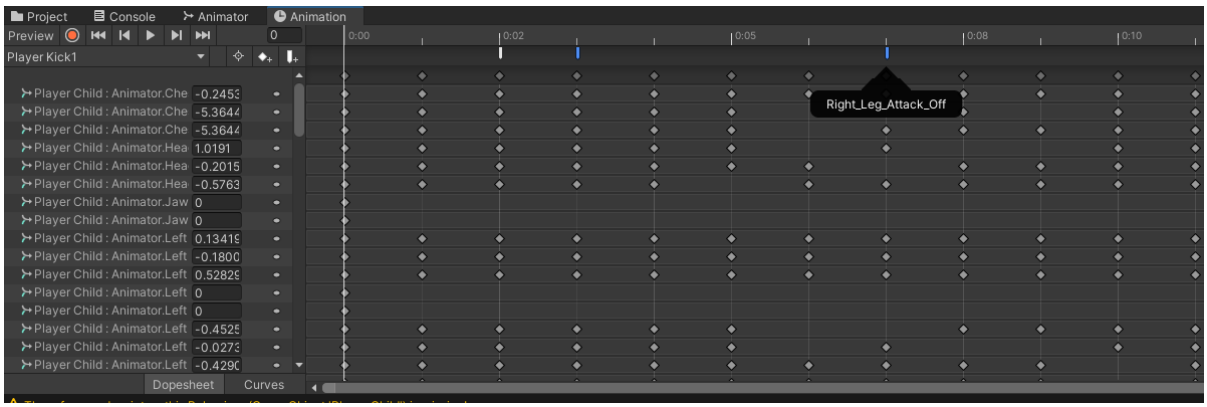

Figure 21 Right_Leg_Attack_Off event

We then added visual effects to the character's attacks to make the action scenes look better. In addition, I also added a random mechanism when the character performs a complete combo that will increase the rate to cause the enemy to be Knocked out for a period of time to create a variety in the way of fighting:

Figure 22 Enemy Knock End Animation

4.1.5. Health Bar

We created the HealthScript class to control the health of the characters and the amount of damage they take. In there is a method called ApplyDamage(float damage, bool knockDown) used to show the remaining health of the character after being hit, if the character runs out of health, the character will die. At the same time, this method also assumes the task of randomly knocking down the enemy when the Player hits it.

Figure 23 Health Background (red)

The Health UI section is on top of and covers the Health Background:

Figure 24 Health UI (yellow)

Finally, we created the HealthUI class to control the short length of the yellow health bar depending on the character's remaining health:

```
void Awake() {
    health_UI = GameObject.FindWithTag(Tags.HEALTH_UI).GetComponent<Image>();
}

public void DisplayHealth(float value){
    value /= 100f;

    if(value < 0f){
        value = 0f;
    }

    health_UI.fillAmount = value;
}
```

Figure 25 HealthUI class Codes

4.1.6. Sound

Of course, we can't lack the sound in a fighting game, right? To be able to control the audio we have created some methods inside the CharacterAnimationDeligate class as follows. Attack_FX_Sound() is the method that shows the sound when the Player hits, CharacterDiedSound() is the method that makes the sound when a character dies, Enemy_KnockedDown() is the sound when the enemy is knocked down, Enemy_HitGround() is the sound when Enemy falls down to the ground. At the same time, we also create events similar to the HitBox section inside the animations to mark the time of these audio clips.

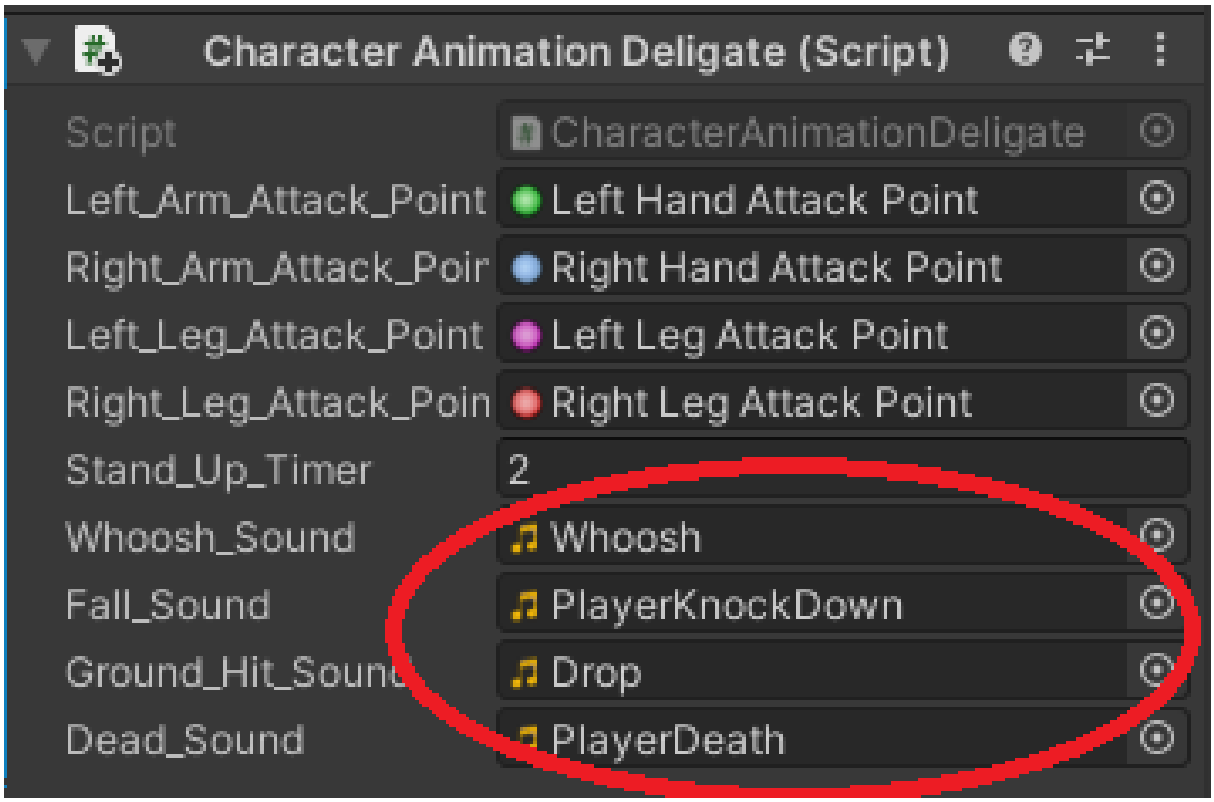

Figure 26 Control Sound Effect Files

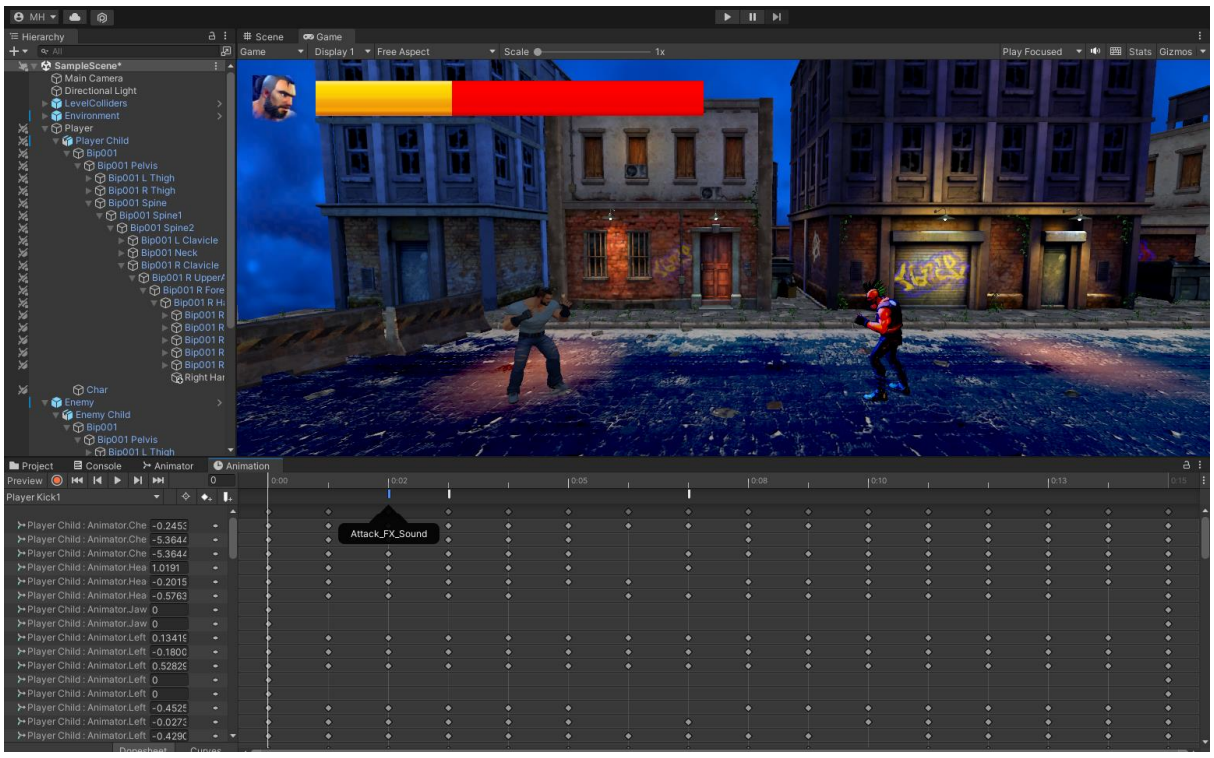

Figure 27 Attack_FX_Sound event

And finally we create an Object that is BG Sound to play the background music of the game with an endless loop:

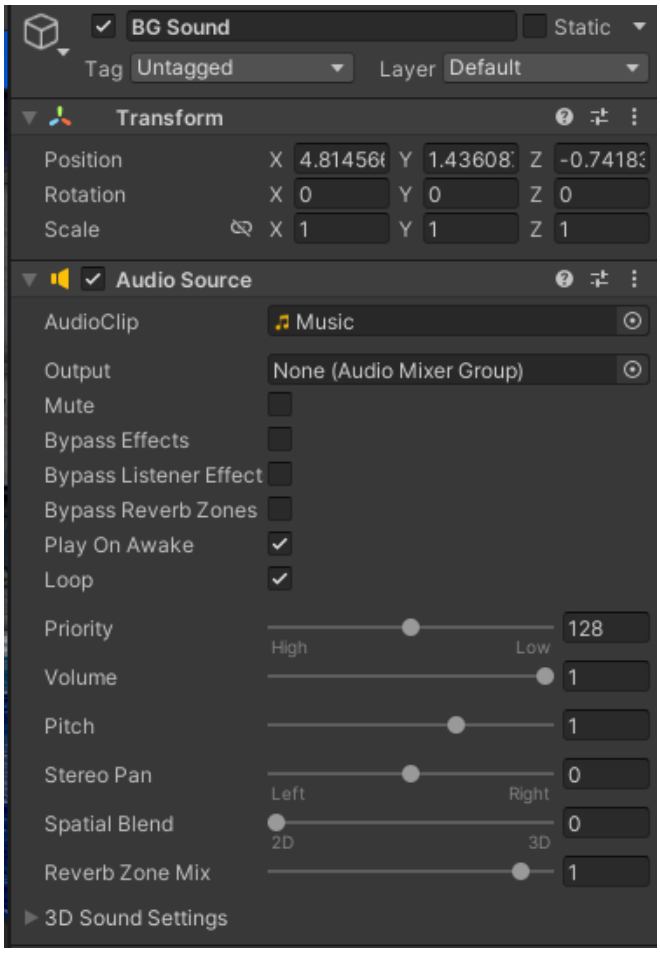

Figure 28 BG Sound Game Object

4.1.7. Camera Shake

Finally, a pretty cool effect we've created and added to enhance the game's experience is CameraShake. We created the ShakeCamera class to do this effect. Among them is the Shake method, which will cause the camera angle to randomly move back and forth in a short period of time and make the player feel like the screen is shaking. We've set up this effect for times when Enemy is knocked down which will take the player experience to the next level.

```
void Shake(){
    if(shouldShake){
        if(duration> 0f){
            transform.localPosition = startPosition + Random.insideUnitSphere * power;
            duration -= Time.deltaTime * slowDownAmount;
        }

        else {
            shouldShake = false;
            duration = initialDuration;
            transform.localPosition = startPosition;
        }
    }
}
```

Figure 29 Method Shake Code

4.2. Results

After a long process of studying and testing above, we have achieved the following results.

4.2.1. Graphics

In terms of graphics, we have successfully set up an environment that gives a realistic sense of the depth of the scenes. The graphics of this project were able to realize the basic potential of the 3D environment. In addition, our character models are also meticulously designed and perfected to create a strong feeling for each character and create a contrast in the styling style and color scheme between our characters. player and Enemy

Figure 30 Final Graphic of Project

4.2.2. Animation

Regarding animations, we have created a lot of different animations for both Player and Enemy. This creates a variety of movements of the characters that make the game even more realistic. In addition, the movements are also made extremely meticulously from each frame to the smallest parts on the character models. All of the above have created very realistic and attractive moving and fighting animations for this game

4.2.3. Combat Mechanism

Next is the result of the game's combat mechanics. Not only are there two ways to attack, punch and kick, but players can also use beautiful and effective combos by pressing the correct key combinations. In addition, we also incorporated random elements of luck into the combat mechanics such as the rate of knock down attacks when successfully performing Player combos. This has created diversity and a unique attraction for the combat mechanism of this game.

4.2.4. Effect

Another indispensable ingredient is the effects. Visual effects when the Player hits Enemy make the action scenes extremely attractive. Sound effects for animations increase the realism and experience of the player. Shake Camera effect makes knock out scenes extremely attractive. All contribute to the perfection of this game

4.2.5. Sound

Finally, sound is an extremely important part that greatly affects the quality of a game. The achievement through the sound achieved after completing this game is extremely suitable. The volume is just enough to make the game not too noisy to affect the player. The sound of each blow is extremely realistic. And especially the heroic but vibrant background music makes the atmosphere in the game like a hot game that contributes a lot to the player's game experience.

CHAPTER 5

DISCUSSION AND EVALUATION

5.1. Discussion

This is usually a Hit em up game in 2D where you play as a hero who walks to the right and kicks away any enemies that stand in your way. The controls tend to be simplified to two buttons. It works pretty much like other fighting games.

5.2. Comparison

We designed this game as a simplified version of Little Fighter 2. The game is simply about moving, punching and kicking your opponent. Does not include counterattack, dodge or defense like Final Fight 3, it has been simplified so that we can design it more easily without losing your enjoyment of the game. However, we use a more eye-catching 3D graphics platform and add visual effects, animations, etc. This has created a very different character for our game.

5.3. Evaluation

We think our game is pretty good. And the graphics are quite realistic, sharp, and vivid. It's still simple, without as many characters as in other fighting games. However, there are still combo moves, full sound visual effects and still very different features compared to some products of the same genre.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1. Conclusion

In the end, we still haven't been able to complete this game completely. But above all, we were able to apply all the knowledge we have learned from Computer Graphics to make this product. Although not completed, all the basic and necessary elements of this game have been completed and just need to complete a few other mustache elements. In short, we are extremely satisfied with a product that has full elements from mechanics, graphics, animation, effects, sound, ... which was completed in less than 3 months while we have not yet completed the work. Learned about Unity or C#

6.2. Future work

In the future we will try to perfect the game and create a few more necessary features such as:
+ More combos
+ Add more characters
+ PvP game mode
+ Online game mode
+ Block mechanism
+ Special Skill Mechanic
+ Recovery mechanism
+ More levels and different difficulty
+ Add more levels
+ The control panel settings of the game, ...

REFERENCES

1. MIKE GEIG, “Unity Game Development in 24 Hours”, Third Edition, 2018
2. JOE HOCKING, JESSE SCHELL, “Unity in Action: Multiplatform Game Development in C# with Unity 5”, Second Edition, 2015.
3. MARK J. PRICE, “C# 8.0 and .NET Core 3.0”, 4th edition, 2019.
4. ANDREW STELLMAN, “Head First C#: A Learner’s Guide to Real-World Programming with C#, XAML, and .NET”, 4th edition, 2013.
5. JESSE SCHELL, “The Art of Game Design: A Book of Lenses”, Third Edition, 2019

APPENDIX

According to a report by Newzoo, the global Game market generates revenue of USD 159.3 billion in 2020, an increase of 9.3% compared to the same period last year. One of the great driving forces behind the strong growth of the Game market in 2020 is the result of the "lockdown" measures of countries during the Covid period.

Besides, the fact that video game publishers simultaneously launched new generation gaming consoles at the end of 2019 is also a reason contributing to the revenue growth of the Game market.

Global Game market revenue by device

Mobile games (including smartphones and tablets) are the largest segment in 2020 with revenue of $77.2 billion, up 13.3% year-on-year. The fastest-growing mobile game ecosystem in the Asia-Pacific, Middle East, and Africa markets. However, the Americas, Europe, and China are also growing strongly. Contrary to Mobile Games, PC Games tend to decrease because many gamers have started to switch to playing Mobile Game genres. PC Game revenue in 2020 reached 36.9 billion USD. The console game segment has grown quite strongly, especially during the peak period of Covid. According to reported data, Game consoles brought in $ 45.2 billion in revenue, up 6.8% over the same period last year.

Revenue level of the global Game market by region

In terms of regions, the games market in the Asia-Pacific region generated $78.4 billion in revenue, up 9.3% year-on-year and accounting for nearly half of the total game revenue worldwide. demand in 2020. The Middle East and Africa region achieved the fastest growth compared to the same period last year, increasing by 14.5% compared to 2019.

All video game segments saw an increase in engagement and revenue. In, mobile games have the largest increase (up more than 13.3% over the same period in 2019). Up to the present time, mobile games still have the largest market share in the game industry, and in the future, it is difficult to be "usurped" by other markets thanks to the advantages of convenience, ease of play, easily accessible to a wide range of users because today, most people use smartphones, and especially the cost of paying for mobile games is much lower than for console or PC game genres. It is estimated that in 2020 there will be a total of 2.7 billion Game players globally.

The Asia-Pacific region generates revenue from Games of USD 78.4 billion, accounting for 49% of the global games market, a year-on-year growth of 9.9%. North America is the second largest region in terms of revenue from video games, accounting for a quarter of the total global Games market in 2020 (equivalent to approximately $40 billion). This figure represents an increase of 8.5% year-on-year, the second-highest year-on-year growth rate compared to other regions. With a growth rate of 7.8%, Europe became the third largest developed game market, with a revenue of 29.6 billion USD, accounting for about 1/5 of the global game market.

Europe and North America are the two pioneering Game markets of the global Game industry. Therefore, it is easy to understand why the 2020 revenue of these two markets is lower than that of other emerging markets.

By the end of 2020, the number of Gamers globally is predicted to be about 2.7 billion, an increase of more than 135 million compared to the previous year. In, the number of gamers in the Asia-Pacific region is 1.4 billion, accounting for 54% of the total number of players worldwide. East Asia, especially China, has the highest percentage of gamers in the Asia-Pacific. The North American region has the lowest player ratio of all regions with a modest player count of about 14% of the worldwide player base.

COMPUTER GRAPHICS

Computer graphics is an area of computer science that deals with the mathematical foundations, algorithms, and techniques that enable the creation, display, and manipulation of images on a computer screen. Computer graphics is related to a number of fields such as algebra, analytic geometry, graphic geometry, optics, ... and computer engineering, especially hardware manufacturing (of all kinds). monitors, input and output devices, graphics cards, etc.).

In a broader sense, computer graphics is a method and technology used to convert data and images on a screen by a computer. Computer graphics or computer graphics techniques are also understood as methods and techniques for creating images from mathematical models that describe objects or data derived from real-life objects. The term "computer graphics" was proposed by an American expert named William Fetter in 1960. At that time he was working on building a model of an airplane cockpit for Boeing. William Fetter has based on 3D images of the pilot model in the cockpit to build the optimal cockpit model for Boeing aircraft. This was a very new research method at that time. This method allows designers to visually observe the driver's position in the cockpit. William Fetter named his method computer graphics...

The history of computer graphics in the 1960s is also marked by the SketchPad project developed at the Massachusetts Institute of Technology (MIT) by Ivan Sutherland. The achievements were reported at the Fall Joint Computer conference, and this was the first time that people could create, display and change image data directly on a computer screen in real time. This Sketchpad system is used to design electrical circuits and includes the following components:

1. CRT screen
2. Bright pen and a keyboard including function keys
3. A computer contains a program that processes information

With this system, users can directly draw circuit diagrams on the screen through a light pen, the program will analyze and calculate the necessary parameters of the circuit drawn by the user.

Graphics engineering was continuously perfected in the 1970s with the emergence of graphics standards that enhanced the interoperability and reusability of graphics software and libraries.

The rapid development of microelectronics technology and computer hardware in the 1980s led to the emergence of a series of circuit boards that supported graphics access along with a significant reduction in the price of personal computers. Graphics go deeper and deeper into real life.