ALANYA ALAADDİN KEYKUBAT ÜNİVERSİTESİ
RAFET KAYIŞ FACULTY OF ENGINEERING

UNDERGRADUATE THESIS

DEVELOPING A WEB-INTERACTED GAME USING UNREAL ENGINE 5

Ali Nazım KÜÇÜK, Ümit Numan DUMAN, Halil İbrahim ÖZDEMİR

ALANYA, ANTALYA

2023

THESIS APPROVAL

The thesis titled "Developing a Web-Interacted Game Using Unreal Engine 5" prepared by Ali Nazım KÜÇÜK, Ümit Numan DUMAN and Halil İbrahim ÖZDEMİR has been unanimously/majority accepted as a Bachelor's Thesis on 22/06/2023 by the following committee members at Alanya Alaaddin Keykubat University, Rafet Kayış Faculty of Engineering, Department of Computer Engineering.

Supervisor: Dr. Özge ÖZTİMUR KARADAĞ

Jurist:

Foreperson: Doç. Dr. Alper Kürşat UYSAL

Member: Dr. Özge ÖZTİMUR KARADAĞ

Member: Dr. Yılmaz Kemal YÜCE

Member: Dr. İbrahim Rıza HALLAÇ

Member: Dr. Kübra UYAR

MORALITY

I declare that all the information in this thesis, prepared in accordance with the thesis writing rules of Alanya Alaaddin Keykubat University Rafet Kayış Faculty of Engineering, is accurate and complete. I affirm that scientific ethics were adhered to during the generation of the information, and I have indicated all the sources I have benefited from by making proper citations.

22/06/2023	**22/06/2023**	**22/06/2023**
Ali Nazım KÜÇÜK	Ümit Numan DUMAN	Halil İbrahim ÖZDEMİR

ABSTRACT

Undergraduate Thesis

DEVELOP A WEB-INTERACTED GAME USING UNREAL ENGINE 5

Ali Nazım KÜÇÜK - Ümit Numan DUMAN - Halil İbrahim ÖZDEMİR

Alanya Alaaddin Keykubat University
The Faculty of Engineering
Department of Computer Engineering

Supervisor: Dr. Özge ÖZTİMUR KARADAĞ

This thesis project presents a game developed with Unreal Engine 5, featuring QR codes strategically placed throughout the virtual environment. Scanning these QR codes redirects players to a dedicated website offering unique hints and guidance for each code encountered.

The game aims to engage players in an immersive adventure, requiring them to solve puzzles and overcome challenges. The QR codes serve as an interactive tool, providing specific clues related to the encountered obstacles.

A dedicated website associated with the QR codes acts as a central hub, offering comprehensive information and tailored hints to enhance players' understanding of the game.

The development process involved utilizing Unreal Engine 5's advanced graphics and design tools to create a visually stunning and interactive virtual environment.

This project demonstrates the integration of QR codes into a game developed with Unreal Engine 5, providing an innovative and immersive gaming experience. The dynamic hint system allows players to explore and overcome challenges while maintaining an engaging gameplay experience.

2023, 43 pages
Keywords: QR Code, Unreal Engine 5, Game, Storytelling

IMAGES INDEX

TABLE OF CONTENTS

1. INTRODUCTION

This thesis explores the development of a game using Unreal Engine 5, incorporating strategically placed QR codes within the virtual environment. These QR codes serve as interactive elements that, when scanned, direct players to a dedicated website offering unique hints and guidance for each code encountered.

The aim of the game is to provide players with an immersive adventure that challenges them to solve puzzles and overcome obstacles. By utilizing QR codes as an integral part of the gameplay, players can access specific clues related to the encountered challenges, enhancing their problem-solving skills and overall engagement. The dedicated website associated with the QR codes acts as a central hub, providing comprehensive information and tailored hints to assist players in progressing through the game. This feature allows players to access additional support and gain a deeper understanding of the game's mechanics and objectives.

The development process involves leveraging the advanced capabilities of Unreal Engine 5, a powerful game development platform renowned for its stunning graphics and intuitive design tools. The virtual environment created for the game is visually captivating, aiming to draw players into an immersive and visually appealing gaming experience. By integrating QR codes and implementing a dynamic hint system, this thesis project aims to offer an innovative and engaging gaming experience. The strategic placement of QR codes throughout the game world, combined with the dedicated website's comprehensive support, ensures that players can enjoy a challenging yet rewarding gameplay journey.

Through this thesis, we explore the potential of QR codes and Unreal Engine 5 in creating a captivating and interactive game, providing players with a unique and immersive gaming experience.

1.1 WHY UNREAL ENGINE 5.1 INSTEAD OF OTHERS

Unreal Engine 5.1 stands out as the premier choice among game developers due to its unparalleled power and versatility. One of its most significant advantages is its

unrivaled graphical capabilities. With the introduction of Nanite, a virtualized micropolygon geometry system, Unreal Engine 5.1 can handle vast amounts of high-resolution assets without compromising performance. This breakthrough technology allows for incredibly detailed and lifelike environments, bringing games to new levels of realism. Furthermore, the inclusion of Lumen, a dynamic global illumination solution, empowers developers to achieve stunning lighting effects in real-time, enhancing visual fidelity and immersion.

Additionally, Unreal Engine 5.1 offers a comprehensive suite of developer tools and features. The Blueprint visual scripting system enables both experienced developers and newcomers to create complex game mechanics and interactions without the need for extensive coding knowledge. This accessibility makes Unreal Engine 5.1 an ideal choice for indie developers and small teams. Moreover, the engine supports a wide range of platforms, including PC, consoles, and mobile devices, providing developers with the flexibility to target multiple platforms and reach a broader audience.

Another compelling aspect of Unreal Engine 5.1 is its active and supportive community. With a vast and dedicated user base, developers have access to a wealth of resources, tutorials, and forums where they can seek assistance, share knowledge, and collaborate. This vibrant community fosters innovation encourages the exchange of ideas and ensures that developers can overcome challenges and make the most of the engine's capabilities.

Furthermore, Unreal Engine 5.1 offers a generous business model. The engine is free to use, with royalties only due when a project reaches a certain level of commercial success. This accessibility lowers the barrier to entry for aspiring developers and independent studios, enabling them to bring their creative visions to life without significant upfront costs.

In conclusion, Unreal Engine 5.1's combination of cutting-edge graphical prowess, user-friendly tools, robust platform support, thriving community, and advantageous business model solidify its position as the most powerful and appealing game engine in the industry today.

2. LITERATURE REVIEW

2D games are the oldest in the book, The Pong Game(1972) is the very first mainstream game, and it was 2D. One of the most popular games in history is obviously Mario, the first Mario published in 1985 so we can say 2D games are the ancestors of modern ones. As the time goes on, 2D games shaped with 3D graphics for instance: Donkey Kong Country: Tropical Freeze, Super Smash Bros. Ultimate, Metroid Dread. Our game has 2D gameplay and 3D graphics for having depth and atmosphere. We decided to give the player a hitting story, nice graphics and easy gameplay. There are lots of games like us in the sector, our game is similar to This War of Mine, Beholder, Limbo, Hollow Knight, Braid, Mark of the Ninja, and Deadlight.

The main genre of our game is Action-RPG. The game will contain combat mechanics and action sequences. It also includes RPG elements in the game we developed. Thus, players will be able to enter into dialogue with other **NPC** and get detailed information about the characters and the world of the game. Some of the games that use this genre are Baldur's Gate 3, S.T.A.L.K.E.R, Mount & Blade Bannerlord 2, Fallout.

3. GAME MECHANICS

Our side scroller action-RPG and puzzle game, Revenge: The Dagger, was developed on Unreal Engine 5.1. The game features a variety of mechanics, totaling over 20, including basic movement mechanics and puzzles. The main mechanics that govern the character's movements consist of walking, running, bending, crouching, and jumping mechanics.

3.1 BASIC MECHANICS

To create the basic movement mechanics, we utilized freely available animations from Mixamo.com. Although these animations are compatible with Unreal Engine 4, they encountered issues with the character skeletons due to the updated release of Unreal Engine 5. To overcome this problem, we employed a third-party program called Mixamo Converter,

specifically developed to address these compatibility issues. This program allowed us to seamlessly import Mixamo animations into Unreal Engine without any complications.

We employed the locomotion system provided by Unreal Engine 5 to implement the basic movement mechanics. The programming aspect was achieved using Unreal Engine 5's blueprint system. As our game is a side-scroller, we utilized the 1D Locomotion system, which is a component of this system, designed specifically for side-scrolling projects. Its distinctive feature is providing movement in only two directions. Upon enabling the Locomotion 1D, horizontal motion was selected as the primary motion, with its value restricted between 0 and 650. We added a stop animation for the character at the 0 position, a running animation at the 500 position, and a faster running animation at the 650 position. Once the desired walking animations were integrated into this system, we created the Animation Blueprint for the main character. Subsequently, we added the character's reference to the Event Graph of this blueprint and accessed the character's vector length by connecting it to the VectorXY function. This allowed us to obtain the character's velocity in the X and Y planes. We then created a variable called 'Speed' and stored the speed values in the X and Y planes within this variable. This variable was also linked to the animation update function. Moving forward, we proceeded to the 'Animgraph' section, where we established a State Machine named Locomotion. Within this Locomotion State Machine, we created two additional State Machines named Idle and Run. These two state machines were connected in sequence, enabling the character to transition between the idle and running states. In order to facilitate these transitions, conditions were set. The condition for transitioning to the Idle state was 'Speed ≤ 0,' while the condition for transitioning to the Run state was 'Speed > 5.'

Furthermore, we incorporated a 'Shift key' in the character's blueprint to activate the running function. This key was implemented using the Advanced Input System provided by Unreal Engine 5.1, and the necessary key assignment was made in the Input Map. When the Shift key is held, the value was set to 650, and when released, it was set to 500. To enable the character to bend and move while crouching, we created an 'isCrouch Boolean' variable in the character's blueprint. After setting up the Crouch input and assigning it in the Input Map, we called the CTRL key in the character's blueprint. When the key is pressed, the 'isCrouch Boolean variable' is set to True, and when released, it is set to False. To make the

character's capsule component shrink during crouching, we activated the 'Can Crouch' feature in the character movement. Depending on this feature, the Crouch function was connected to the True side, and the Uncrouch function was connected to the False side, utilizing the isCrouch Boolean variable.

Similar to the Walking/Running mechanics, we imported bending idle and leaning walking animations from Mixamo using the Mixamo Converter. The same procedures and techniques used for Walking/Running were employed for the crouching mechanics as well. Additionally, we accessed the isCrouch variable created in the Event Graph of the Animation Blueprint and connected it to the additional Crouch Boolean in the Event Graph. This Boolean was then linked to the Animation update function. On the Animgraph page, we established a Crouch state within the Locomotion state, and both the idle and walk states were connected bidirectionally to the Crouch state. The conditions for these connections were identical to those used for walking and stopping animations. Moreover, an 'AND' gate was included in the connections to the Idle and Walk states, along with a note, which called the Crouch boolean2 created in the Event Graph of the Animation Blueprint. This ensured that the animations were seamlessly incorporated into the character's movements without encountering any errors.

For the jumping mechanics, we created a new input and assigned the Space key in the Input Map. This key was then implemented in the character's blueprint, connecting it to the JUMP function when pressed, and the Stop Jumping function when released.

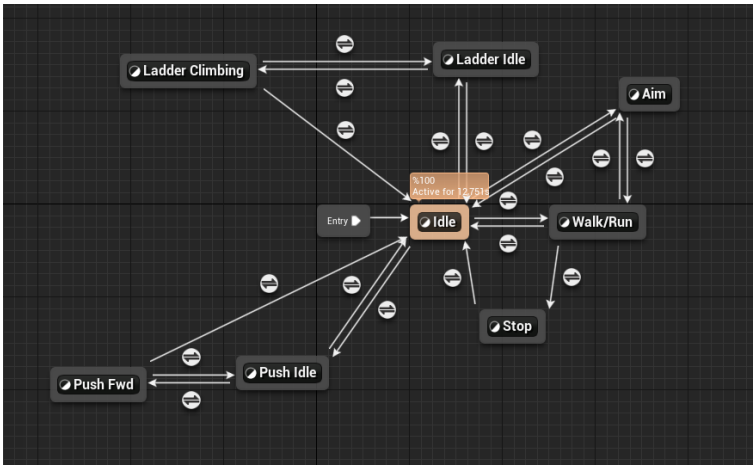


Image 3.1

3.2 PARKOUR MECHANICS

In addition to the basic mechanics, the main character in Revenge: The Dagger game features various advanced movement mechanics, including sliding, vaulting, climbing stairs, climbing on objects, moving laterally while climbing objects, and zipline mechanics.

3.2.1 SLIDE MECHANICS

To implement the sliding mechanic in the game, we obtained an animation from Mixamo and imported it into Unreal Engine 5.1. In order to ensure that the character's motion remains in a fixed plane, we enabled root motion by clicking on the animation. Root motion allows the animation to control the character's movement within a specific plane. If this option is not enabled and the animation used involves the character traveling a distance, the animation will play independently of the character's capsule component. To address this, we created a separate animation montage for the sliding motion by right-clicking on the animation and selecting "Create Animation Montage."

To activate the slide mechanic, we created an input and assigned it to the 'C' key in the Input Map. Once the assigned key is called in the character's blueprint, we added a branch to check for the input. While the character is playing the animation montage, we called the vector length from the character's movement to determine the distance the character should travel. We then added a condition 'greater than 230' to ensure that the character can only perform the slide while in motion. This condition was connected to an AND gate, along with a check to determine if the character is in the air or not. The falling boolean variable was connected to the NOT boolean and then linked to the AND gate. This branch was connected after the AND gate.

Following the branch, we used the 'Set Capsule Half Height' function to reduce the height of the character's capsule component, allowing them to pass through low spaces. This enabled the character to slide under obstacles. Additionally, we played the animation montage created earlier using the 'Play Anim Montage' node. Through trial and error, we

determined the optimal value of 30 for adjusting the character's position on the Z-axis, ensuring they remain close to the ground during the sliding animation.

To enable the character to stand up and resume walking after the sliding animation concludes, we reversed the operations performed during the animation. Instead of subtracting 30 from the character's location on the Z-plane, we added 30 to it.

3.2.2 VAULT MECHANICS

For the vaulting mechanics, we incorporated a vaulting animation into the game and created an animation montage for it. Additionally, we established an input and assigned it to the ALT key in the Input Map. Within the character's blueprint, we introduced a boolean variable called 'CanWarp' and created three vector variables named VaultStartPose, VaultMidPose, and VaultLandPose, which store the character's positions during vaulting.

After adding the MotionWrapping Component to the character, we called the Vault key in the blueprint. This key was bound to a custom event called Vault. Subsequently, we connected a branch to the created vault event and set 'CanWarp' as the condition for the branch. This allowed us to determine the maximum height at which the character can vault onto objects.

To facilitate the vaulting process, we set the character's movement mode to flying and disabled collision to prevent the character from getting stuck during the vault. Within the vaulting animation, we added three notifies: StartWrap, MidWrap, and EndWrap. These notifies were positioned at the beginning, during, and end of the animation, respectively.

In the character's blueprint, we called 'Add or Update Warp Target' and connected it consecutively to the three notifies. The VaultStartPose, VaultMidPose, and VaultLandPose variables were assigned to the 'Warp Target Location' sections of these nodes, respectively. We retrieved the character's rotation using 'Get Actor Rotation' and connected it to the 'Warp Target Rotation' section. The MotionWrapping component was linked to the target part of the nodes.

Next, we invoked 'Play Montage Mesh' to play the vaulting animation and selected the created vaulting animation montage. To ensure the character switches back to walking mode after the animation, we called 'Set Movement Mode' and set it to 'walking' mode. The character's collision was then re-enabled.

Finally, we set the 'CanWarp' Boolean to false and used the 'VaultLandPose' vector variable to ensure the character lands properly. The Z-axis value of the 'VaultLandPose' vector variable was set to 20000.

3.2.3 CLIMB MECHANICS

For the climbing mechanics, animations were first transferred into the game. In this mechanic, it was necessary to create an additional trace channel since the character did not want to climb every wall, and with this additional trace channel, the character would be able to climb only the objects with that trace channel selected. For this, we went to Project Settings from the Edit section of the game engine and from there, the trace channel named 'LedgeTrace' was added by clicking the 'New Trace Channel' button on the page opened by selecting the Collision section and the default was set to 'ignore'. Thus, this trace is set to not be selected by default in each object. Climbing mechanics consists of four parts. These are Grab Ledge, Move Left, Move Right and Exit Ledge.

A Blueprint interface named 'Climb Interface' was created in the content files of the game, so that the functions made could be displayed in the animation blueprint. In this interface, functions named Cangrab, climbing ledge, moveleft/rightclimb were created and boolean variables related to their own fields such as CanGrab, isClimbing were added to their inputs. Then the character blueprint page is opened, and the created interface is added here. A 'forwardtrace' function has been created to determine whether the character is climbable or not, 'height trace' to understand whether the height of this object is climbable, and the 'Move in Ledge' function to control whether the character can move left and right along the edge when holding an edge. These functions are generally used to get the status of the relevant object by sending a sphere trace of the sphere trace character. Forward Trace shoots a beam forward, while Height Trace shoots an upward beam to detect height. On the other hand,

Move in Ledge trace detects whether the direction to go is empty by creating capsule components to the right and left of the character after the character grabs the edges, thus preventing the character from going right or left forever. When the character grabs and presses the spacebar so that he can climb onto the object, the movement mode of the character is set to flying and after the movement is finished, it is set to walking again. Finally, at the end of the algorithms of these movements, the functions created in the climb interface are connected to the relevant movements and set as true. Then the animation blueprint page for animations was opened and the climb interface was added here. After the functions in the Climb interface are called, a variable is created for each of these functions to get their state. Then, in the animgraph, Jump was connected to the part where the animation was one-way, and the isHanging boolean, which checks whether the character grabledges or not, was added as a condition to this connection. Afterwards, a connection was established between all climbing movements, and the boolens and conditions created for their own movements were created in these connections.

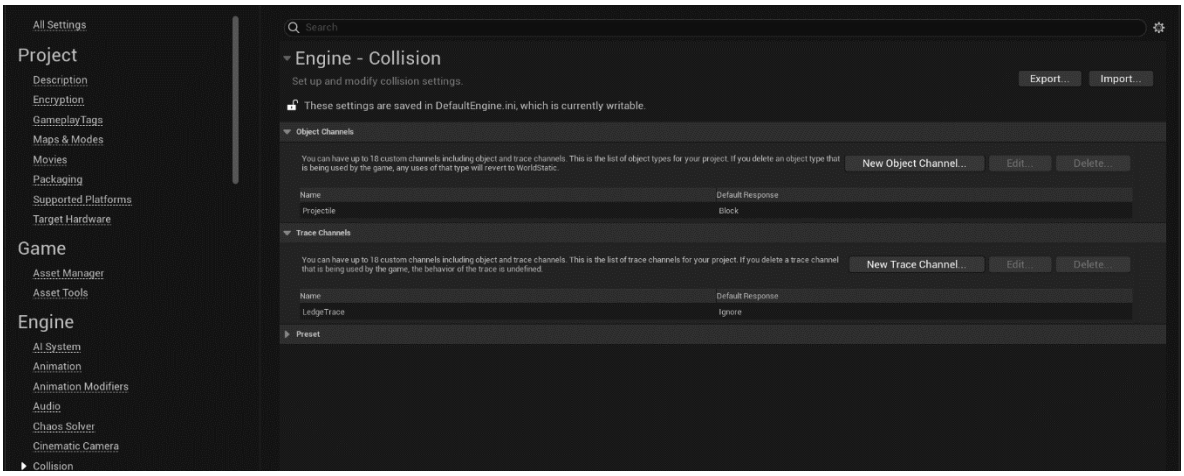


Image 3.2

3.2.4 CLIMB LADDER MECHANICS

In the stair climbing mechanics, after the animations were added to the game engine, a ladder asset was used over Quixel Bridge. After a blueprint of this ladder was created, a

box collision was added to see if the character was in front of the ladder. Afterwards, a function called climb ladder was created in the character's blueprint, where the movement mode of the character was set to flying when the character was inside the box collusion. Adjusted the Z-axis rotation of the ladder so that the character's face is facing the side of the ladder, and the distance between the ladder so that the character does not enter the ladder.

Finally, two Booleans named 'Near Ladder' and 'OnLadder' are created. One of these Booleans OnLadder has been added to the beginning of the function called Climb Ladder, so that it is prevented from automatically exiting when the character passes by a ladder. An input for the interaction is created and assigned to the 'E' key in the input map. Then, after the interact button was called in the blueprint, a node named climbladder was created and this node was connected to the interact button. A branch has been added in this node. NearLadder Boolean is connected to the branch as a condition. Thus, the character is adjusted so that the ladder can only be climbed when pressing E while close to the ladder. Then a flip/flop was added to this branch. In case A of Flip/Flop the character's OnLadder Boolean is set to True. In case B of Flip/Flop, OnLadder Boolean is set to False. For the animation of the Climb Ladder, the Boolean 'OnLadder' is called in the event graph and bound to the Boolean 'OnLadderAnim'. Then, after entering the locomotion state machine in the animgraph section, two state machines named ClimbLadder and LadderIdle were created. After the two are bidirectionally connected to each other, LadderIdle is bidirectionally linked to the standard idle. After the animations are assigned, the condition is set if OnladderAnim is True and Speed is greater than 5 on the connection from Ladderidle to ClimbLadder. The same is done with OnLadderAnim being True in the section from ClimbLadder to LadderIdle. In the connection from LadderIdle to the standard idle, only the OnLadderAnim is False. Checked the OnLadderAnim's true status in the connection from Standard Idle to LadderIdle.

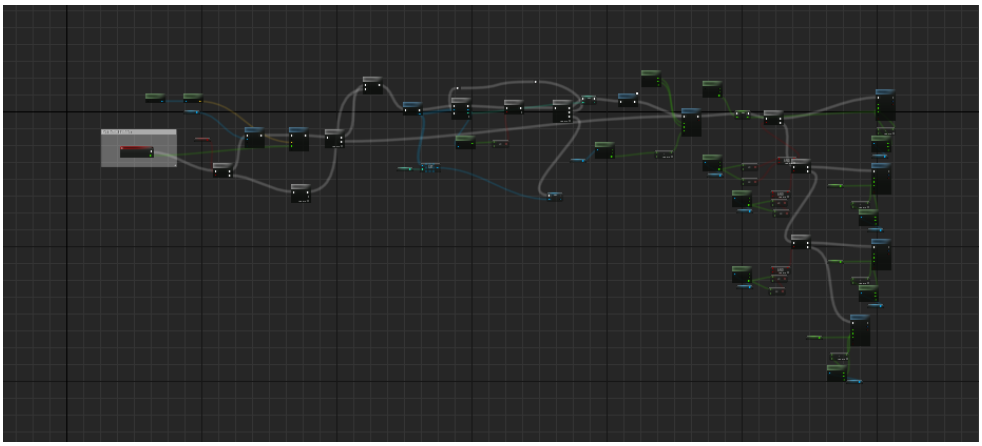


Image 3.3

3.2.5 ZIPLINE MECHANICS

After transferring the animation for the zipline system to the game engine, a direct asset was added to the game. Then a blueprint of this asset was created. A box collusion, a cable for the cable of the zipline, and two static meshes were added to the blueprint to see if the character is near the zipline. In order for the zipline's cable to stop dynamically between two ziplines, a socket was added to the pole asset and the cable was attached to the locations of these sockets. Two vectors were created for the end point and the start point. By making the end point's vector editable and show3dwidget, the zipline was made movable separately from the start point. Then we went to the blueprint of the character and after creating a Boolean called 'playerdetected', it was set to true when we entered the box collision in the zipline's blueprint. Then the branch was connected to the interact button and the generated Boolean was entered here as a condition. The movement mode of the character was set to flying. Right- and left-hand sockets have been created on the character's skeleton. These sockets were used to get the character's hands on the cable while on the zipline. Here, with the trial-and-error method, it was seen that a better image was obtained when the location was 100. Then, the animation of the character going with the zipline was converted to the montage with create montage and added to the blueprint with play montage. In order for the

animation to end exactly at the endpoint, the stop animation function is called while the character is at the endpoint. Then the movement mode was set as walking again

3.3 HEALTH SYSTEM

A widget was created in the content file for the can system, after the screen size was taken with the canvas panel in this widget, a health bar was created by adding the progress bar. This bar is assigned as variable. Afterwards, 2 float variables were created in the blueprint of the character. The first one was holding maximum health, while the other was holding Current Health.

A function named Decrease health was created, damage was assigned to the input of the function as a float variable, and 'Player is Dead' boolean variable was assigned to the output of the function. A branch is connected to this function, then a condition that checks whether the current health is greater than 0 or not, and if it is small, it is directly connected to the output whose Player is dead is set to true. If True, it is connected to Set Current health, the value of set current health is determined by subtracting the damage value from the function's input from the instant value of the current health. Then, Set Current Health is divided by Max Health, and the percentage of the progress bar created in the widget. Afterwards, a branch was added again, and it was checked whether the current health is 0. If current health is zero, player dead becomes True, otherwise it remains false.

Then, Event AnyDamage was called to the blueprint of the eventgraph of the character and connected to the decrease health function, the damage input created was connected to the output of this event, and the generated player dead output was connected to the function of a custom event called branch Die after connecting to the branch. When the die event runs, the inputs are disabled and ragdoll.

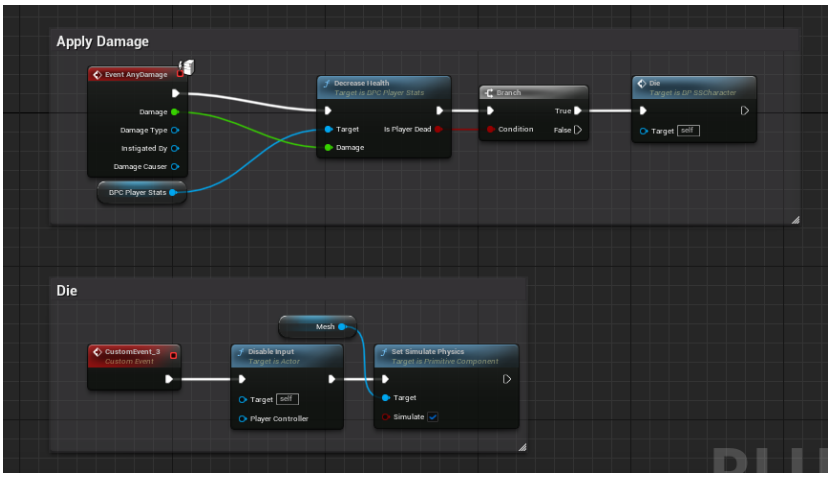


Image 3.4

3.4 AMMO SYSTEM

In the ammo system, an input was created to change the magazine and this input was connected to the input map. Then, one ammo and one current mag integer variables were created, similar to the health system. The function named use ammo was created and one was removed from the current mag in each fire, using clamp integer to set the maximum number to 30 and the minimum number to 0. The result obtained with the subsequent subtraction calculations was assigned to the current mag number.

3.5 COMBAT MECHANICS

3.5.1 ASSASSINATION MECHANICS

Assassination mechanics: Assassinate animations were downloaded first for this. While one of them is the main character's assassinate animation, the other is the animation of dying for the enemy. These animations have been added to notify names to work harmoniously. After creating a blueprint for the enemy, a sphere collision has been added

behind the enemy so that the character will assassinate when he enters this collusion. Created a skeletal mesh referencing the character and added to the back of the enemy. After the location and rotations of this reference character were taken, these values were assigned to the main character's location and rotations when assassinated. When the Assassinate operation occurs, the function that is created on the enemy character and makes the enemy ragdoll is called.

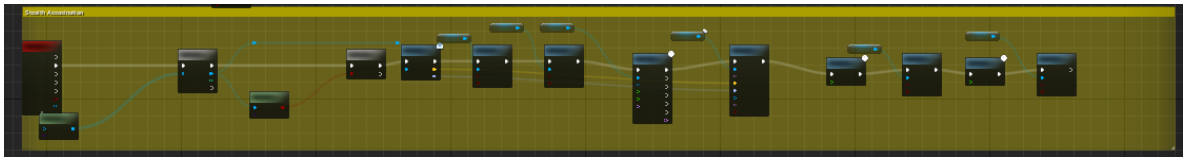


Image 3.5

3.5.2 WEAPON MECHANICS

A rifle and gun were added to the game engine, then a socket was created for each weapon in the character's skeleton to adjust the positions of these weapons. The weapons were sized here. A sphere was used as a reference point for the exit position of the bullet. A blueprint was created for each of these weapons. In general, if these blueprints press the interaction key when they enter the character box collusion, the weapons allow them to attack the sockets that have been created. A rifle store socket was also created for the rifle. Thus, the character was able to take his rifle on his back. For the bullet, one projectile was created for both the rifle and the pistol. After determining the projectile movement here, a tag called Enemy was created and added as a branch so that the bullets only damage the enemy. This tag has also been added to the blueprint of enemy characters. In order for the blood effect to appear on the person hit by the bullet, the impact position of the bullet was taken, and the blood effect was spawned there. Afterwards, the apply damage function was called to determine the damage of the bullet. Random integer was used, and 4 numbers were entered so that the bullet did not have a fixed damage. The system was allowed to randomly select one of these numbers and assign it as a damage value. Finally, the bullets were destroyed by destroying the actor. These operations were also done for enemy characters, after writing 'Damageable' in the tag part, the same tag was given to the main character, allowing the main

character to be damaged by enemy bullets. Then an input for the left mouse button is created and called on the character's blueprint. When this function is called, the projectile exits the reference point. Animations for rifle and pistol have been lowered as the character requires a different locomotion when moving with rifle and pistol. Then, after these animations were adjusted with 1D locomotion as in the character motion system, the animation was blueprinted. Here, after loading the state machine with standard locomotion, it is possible to switch between blend pose and two locomotion’s. The condition here is set to 'GetHandgun' with 'GetRifle'.

An aim offset was created for the aim system. Then, an animation aiming up and down was duplicated and duplicated. The unnecessary parts of these duplicated animations were only pointed up and only down. After throwing into the aim offset, the pitch value of the character looking up was set to 90, and the pitch value looking down was set to -90. Afterwards, a state machine named 'aim' was created in the animation blueprint of the character and connected to the idle and walk state machines and AimOffsets were thrown into it. The 'isAiming' condition was created as the condition.

Then, the function that follows the mouse position in the blueprint of the character and assigns the pitch value according to the position is made. This function is set to the input key that works with the right click button of the mouse, and 'isAiming' is set to True when right clicked, and false when the key is released.

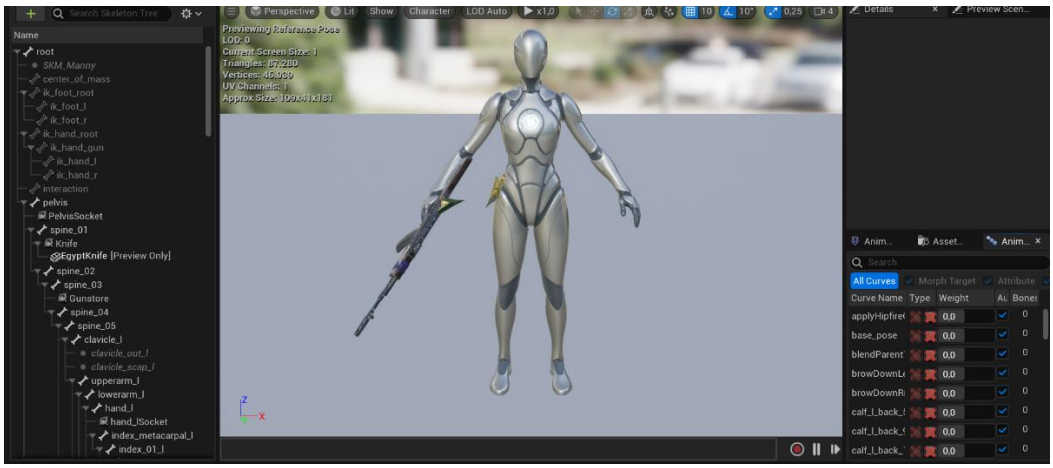


Image 3.6

3.5.3 CLOSE COMBAT

First, attack and damage taking animations were transferred to the game engine. Then it was tagged with notify and it was made to work in harmony.
Developed for both the enemy character and the main character. Reference points were placed in front of the characters for kicking and left and right hands for punching. Each time the characters attack, they are allowed to hit in the direction of the traces coming out of these reference points. In order for this to appear as a combo, it was aimed to create a fluid combat system by enabling it to switch to the other animation in each attack.

3.5.4 ARTIFICIAL INTELLIGENCE

3 artificial intelligence enemies were used in the game. The first of these is close combat artificial intelligence. These enemies try to detect the player by seeing and hearing. When they detect the player, they try to destroy the player by using melee mechanics.
The second type of AI is armed AI, these enemies are still looking for the player like melee enemies and trying to kill them when they find them. However, in addition, these enemies are looking for an object to take cover by scanning nearby objects, and when they find it, they are protected from the bullets of the main character and can shoot from the cover.
The third type of AI is the enemies that are stationary in place. When these see the player, they start shooting at him, but they do not move. These were used on Snipers and turrets.

3.6 DOOR MECHANICS

1. Standard Door: This is the door that opens when the player comes to the door and presses the interaction button.

2. Pin Door: When the user presses the interaction button in this door system, an interface opens, and the user tries to enter the correct password here. If it is entered incorrectly, the door will not open.
3. Keyed door: In this door system, the user is requested to have the key of the door. If the user has the key, they can open the door, but if they don't, they cannot open the door.
4. Puzzle Integrated door: In this door system, after the user solves a certain puzzle, the OpenDoor function of the door works, and the door opens.
5. Bullet Time Mechanic: A power unlocked after the main character acquires the dagger. When Capslock is pressed, it slows everyone except the main character for a certain second. A certain time to reuse after use.

3.6 PUZZLES

1. Library Puzzle: In this puzzle, the player tries to find the missing book, after finding the missing book, he takes this book with the interaction button and puts it back in the library. After putting it in the library, the book is placed in the empty part of the library and the bookcase opens to the side, opening a passage for the player.
2. Bridge Puzzle: In this puzzle, the player is asked to turn the bridge, which consists of three pieces and whose pieces are rotated randomly, with the "Z", "X", and "C" keys to make it passable.
3. Scale Puzzle: This puzzle was prepared based on Anubis' scales to be used in the corn section. The aim of the puzzle is for the player to find the canopic where the mummy's heart is located and the feather, and after putting them on the scales, open the sarcophagus room. The balance asset used in the puzzle was first disassembled in the blender and thrown into the game as such, so that when the puzzle is solved, both ends of the scales are directed to different sides and a feeling of weight is provided.
4. Valve Puzzle: In this Puzzle, the user must try to reach the kaponik by starting from the god symbols on the valves and the poem in the area where the valves are located.

By reading the QR codes on the cards in the pyramid, they can get information about the god from the game's website.

5. Statue Puzzle: In the puzzle in the castle part of the game, the player tries to open the ventilation cover, which is a way to go to the upper floor of the castle, by turning the statues to the right side.
6. Item collection quest: In this optional quest in the Brazil section, the user tries to find 5 items. He tries to get the first clue from the NPC he took the quest from when he finds an item, the second clue when he finds three items, and the last clue when he finds all the items.
7. At the same time, the player can take quests from certain NPCs in other maps. he can get clues that will have knowledge about the map.
8. Box pushing puzzle: The player can reach the places they cannot reach by pushing and climbing certain boxes in the game.

3.7 DIALOGUE SYSTEM

In our dialogue system, the progression of dialogues follows a tree-like structure. Each dialogue is connected to others through branching paths, creating a network of possible conversations. This tree structure allows for non-linear storytelling and diverse dialogue outcomes.

At the root of the dialogue tree, we have the initial conversation or starting point. From there, players can choose different dialogue options or responses, leading to different branches of the tree. Each branch represents a unique path or direction the conversation can take.

As players navigate through the dialogue options, they traverse the branches, moving from node to node in the tree. Each node represents a specific dialogue segment or interaction. The dialogue options available at each node depend on the previous choices and the conditions set within the game.

Branches can converge, where different paths merge back into a single point, or they can diverge, leading to entirely separate conversations or outcomes. This branching structure

allows for player agency and multiple possibilities, ensuring that each playthrough can yield different dialogue experiences.

The dialogue tree's design ensures that conversations feel dynamic and responsive to player choices. It enables us to create complex dialogue sequences, character interactions, and narrative arcs that adapt and evolve based on player decisions.

Overall, the tree-like structure of our dialogue system enhances the depth and replayability of the game, providing players with a rich and interactive storytelling experience. With this way, 30 dialogues created from us.

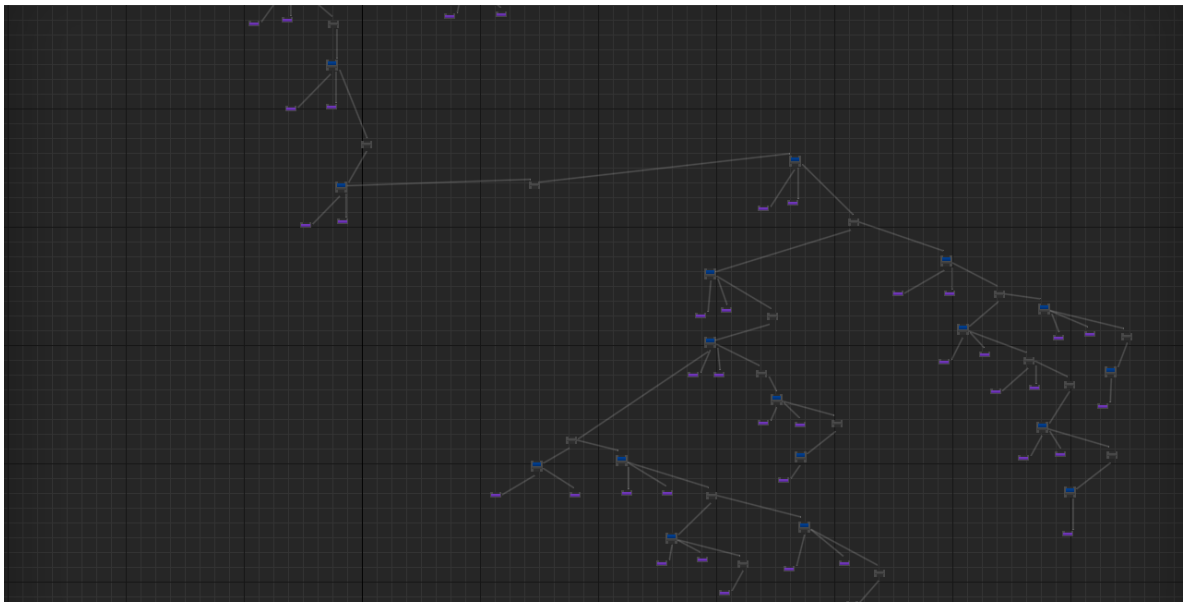


Image 3.7

4. DESIGN AND IMPLEMENTATION

The first chapter of our game, "Revenge: The Dagger," which is part of the "Globetrotter Adventures: Journey of Knowledge," offers users an adventurous experience that promotes research and enjoyable knowledge acquisition.

The story revolves around our character's quest for revenge, taking them on a globetrotting journey. Users will have the opportunity to accompany our captivating protagonist, exploring different countries and embarking on a mysterious adventure.

"Revenge: The Dagger" boasts a captivating storyline and visually stunning graphics, immersing players in a richly crafted world. Our protagonist travels the world, tirelessly pursuing a deadly dagger while fighting against evil forces. Along the way, users will complete various tasks, solve puzzles, connect clues, and progress towards the ultimate goal.

However, the game goes beyond being a mere exciting narrative. "Revenge: The Dagger" encourages users to explore and utilize real-world knowledge. Each country visited by our character is brimming with unique cultures, histories, and landmarks. Users will gain insights into the historical and cultural significance of these places as they journey alongside their character.

Moreover, the game's missions and puzzles are intentionally designed to prompt research and knowledge acquisition. Users will acquire knowledge in various fields such as history, mythology, art, geography, and more. This knowledge will prove valuable not only within the game but also in real life.

"Revenge: The Dagger" aims to entertain and enhance users' knowledge accumulation. Its seamless narrative, immersive visuals, and emphasis on research will captivate players. By embarking on a journey of exploration and knowledge acquisition within an adventure-filled world, users will experience thrilling moments during gameplay while also becoming more knowledgeable individuals in real life.

Remember, "Revenge: The Dagger" endeavors to provide users with an experience that combines both entertainment and knowledge. Your game should be designed in a way that allows users to enjoy themselves while acquiring knowledge.

4.1 CHARACTER DESIGN

Firstly, the method of modifying normal characters using metahumans was utilized. This method involved utilizing metahuman technology to provide us with specific facial shapes, allowing us to adjust and create realistic and humanoid characters. By utilizing

metahumans, we were able to establish visual connections and similarities between characters, enhancing the coherence of the overall design.

Furthermore, fictional characters were generated by deviating from existing metahumans. These characters were enriched with elements of fantasy and imaginative features, allowing for greater creative freedom in their design. After finalizing their appearances, we proceeded to incorporate skeletons and clothing into the characters, making them ready for implementation in the game. This process enabled us to create diverse and visually appealing characters with unique personalities and traits.

Next, a technique called "reality scan" was employed, which allowed us to scan 3D humans. To achieve accurate results, it is recommended to have around 50 to 100 photographs of the subject. The more photographs available, the better the quality of the scan. By utilizing this method, we captured detailed information about ourselves, and the obtained data served as a foundation for character creation.

Subsequently, we utilized platforms like Sketchfab to access a wide range of character assets. These assets, created by various artists, served as a starting point for our character design process. The assets were downloaded and imported into 3D modeling software, specifically Blender, where we could make further modifications and adjustments to align them with our desired character designs.

During the editing process, we focused on customizing various aspects, including the appearance of the characters, their costumes, accessories, and other intricate details. This allowed us to create unique and tailored characters that suited our specific project requirements. The combination of reality scan and asset editing provided us with a versatile and efficient workflow for character design, resulting in visually captivating and individualized in-game characters. In total we created 17 different faces.

Image 4.1

4.2 CREATION OF LEVEL 1

In the first level of our game, we strategically selected Italy as the primary location due to the character's familiar connection to the country through their Italian historian and explorer father. To create a compelling setting, we chose a picturesque bay house as the central hub. We meticulously incorporated elements specific to Italy during that era, such as featuring a classic Italian car commonly seen on the roads. Our research into Italy's cultural and agricultural aspects led us to include sunflower and hazelnut trees surrounding the protagonist's grave within the sunflower field, accompanied by subtle hints intertwining with the storyline.

To immerse the players into the narrative, we paid careful attention to minute details, aiming to enhance their sense of exploration and discovery. We introduced the fundamental movements of the character, allowing users to interact and make decisions alongside the protagonist. By implementing mechanics where players collect cards, we provided an entry point for users to engage with the story, ensuring they recognize the significance of these

moments. Throughout the designed gameplay sections, we strategically placed various obstacles to challenge players and maintain their interest.

Furthermore, we integrated a card system into our map, intending to familiarize users with future challenges they would encounter. As these elements might not be immediately noticeable during active gameplay, we aimed to create a sense of anticipation and intrigue for players, encouraging them to explore beyond the surface level and delve deeper into the game's intricacies.

By meticulously crafting the Italian setting and incorporating engaging gameplay mechanics, we aimed to captivate users and establish a strong foundation for subsequent levels in the game. The attention to detail, combined with the seamless integration of narrative and gameplay elements, seeks to provide an immersive and intellectually stimulating experience for players throughout their journey in "Revenge: The Dagger."

Image 4.2

4.3 CREATION OF LEVEL 2

The second level of our game takes place after a time skip and focuses on the character's development and growth. Following in their father's footsteps, our protagonist undergoes significant personal growth during this time period. In terms of map design, the concept of time and space is crucial. We transition our character from the historical settings of the previous chapter to the modern streets of New York. With this map, we aimed to provide the player with a shift from the tranquility of the first chapter to a more action-packed experience.

Additionally, we wanted to offer the player more freedom by creating multiple paths they can choose from, rather than a linear progression. We provide small pieces of information about these paths through the non-player characters (NPCs) within the map. Furthermore, in some segments, we gauge the player's emotional responses. This aspect of the game highlights its positive influence on the user's behavior.

To capture the ambiance of New York, we incorporated elements such as vending machines, street vendors, trains, enemies, and friendly police forces. We expected the player to uncover a mystery within this setting. We utilized buildings by individually extracting them from a ready-made project called "City Sample." We adjusted the shine of windows and optimized the textures of buildings also we took some cars to drive and then the cars redesigned with changing momentum and their mass. Sidewalks were transformed into tile mode by using models from Sketchfab, and we later applied this tiling method to create asphalt. For pedestrian and other lines, we researched their original sizes and placed them using white plane shapes. We constructed the fire escape step by step in Blender, allowing the character to interact with movable boxes and reach the fire escape. Continuing along this path, we introduced a zipline system and, further on, a relatively easier trail leading to the next level. We believe that exploring this challenging path should be rewarded accordingly. On the other path, we require the player to complete a task in order to discover and enter a pre-determined password.

In summary, our game excels in motivating users toward positive behavior. This section of the game provides a transition in the character's journey, showcasing their personal growth. By immersing players in the vibrant streets of New York, we encourage exploration, uncovering mysteries, and rewarding perseverance. The meticulous design and optimization of various elements, from buildings to textures, contribute to an authentic and engaging experience.

Please note that this text is a thesis-like representation of the information provided and may require further refinement or adjustments based on specific academic requirements.

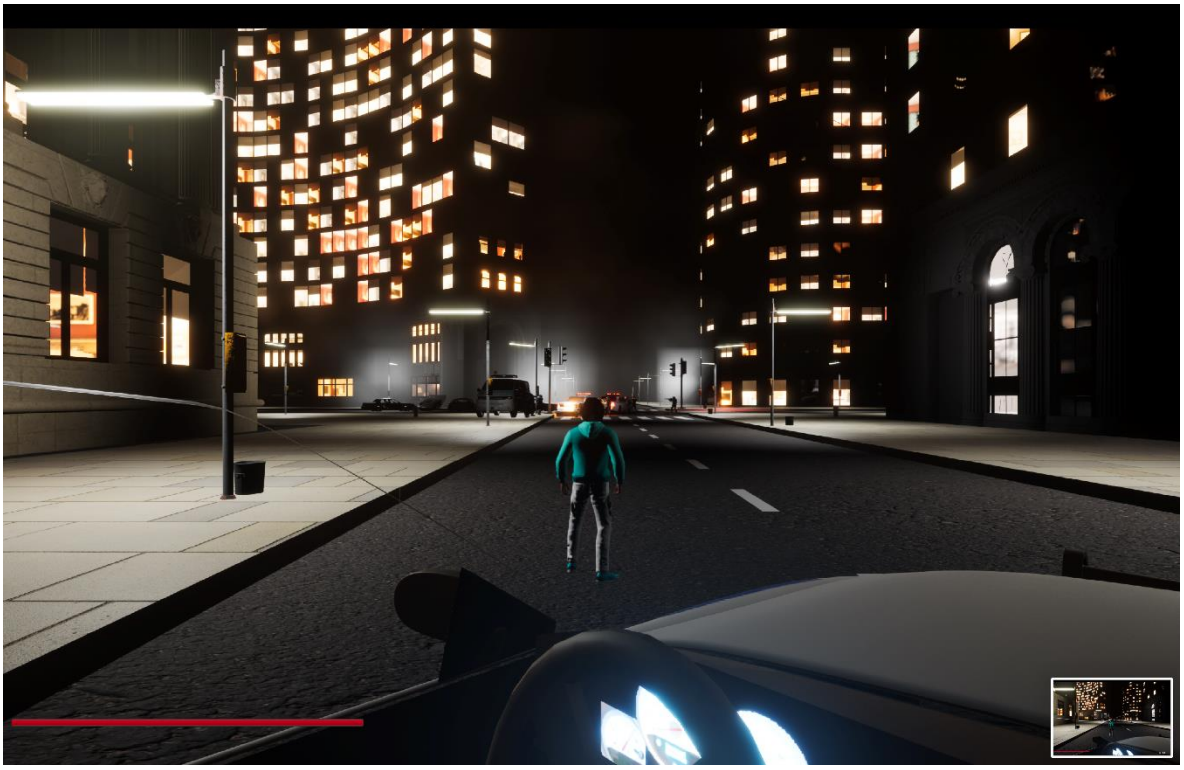


Image 4.3

4.4 TRANSITION LEVEL (HUB)

Furthermore, as a transition level, the character's father's research office serves as a hub for their ongoing investigations. The office is meticulously designed to reflect the atmosphere of a dedicated researcher's workspace, with bookshelves filled with ancient tomes, maps pinned on the walls, and a large oak desk adorned with intriguing artifacts. The character can spend time in this space, delving into the mysteries that lie before them.

As the character progresses in their investigations, they come across various character cards, each representing a unique individual they encounter during their journey. These cards provide valuable insights, knowledge, and potential alliances. By strategically using the character cards, the player can gain access to specialized information, unlock new paths, and forge alliances that aid them in their quest for revenge.

Additionally, the character discovers clue cards, which are vital pieces of information scattered throughout the game. These clue cards offer cryptic hints, hidden codes, or references to historical events that guide the character's progress. The player must carefully analyze and connect the clues to unravel the overarching mystery and move closer to their ultimate goal.

The character's desk serves as a visual representation of their progress and accomplishments. As they collect important items and artifacts during their journey, they can proudly display them on their desk as a personal collection. This feature not only provides a sense of achievement and satisfaction for the player but also offers opportunities for interaction and exploration within the game. Players can examine each collected item, read accompanying descriptions, and potentially uncover additional insights or hidden secrets.

Furthermore, the desk serves as a central point for communication and engagement with the player. It acts as a platform for receiving messages, letters, or important documents that further the storyline. Through these interactions, players can deepen their connection

with the character and gain a deeper understanding of their motivations and the world they inhabit.

Overall, the inclusion of the character cards, clue cards, and the interactive desk feature creates a multi-dimensional gameplay experience. Players are encouraged to think critically, solve puzzles, make strategic decisions, and immerse themselves in the character's journey of revenge and discovery. The combination of gameplay mechanics, engaging storytelling, and interactive elements enhances the player's sense of agency, fosters curiosity, and reinforces the enjoyment of learning and problem-solving.

In terms of design, input from individuals with expertise in visual aesthetics was sought to gather feedback and ensure a visually appealing environment. The office was meticulously crafted, taking into consideration details such as lighting, furniture arrangement, and thematic elements. To optimize performance, a realistic-looking exterior space was showcased, although not directly accessible, using techniques and assets from the Blender program. Curtains were designed using Blender, while floors were created by leveraging assets from the Quixel Bridge program.

To enhance the interactive elements, a box was created around the boards to allow players to inspect them closely when approached from the sides. When inspecting the character cards, a blueprint was implemented to ensure that the image becomes visible as soon as the character picks up the card, utilizing optimized widgets instead of displaying the full FPS view. Similarly, for collectible items, a similar approach was taken, allowing data to be efficiently gathered from all levels using the Game Instance function.

By incorporating feedback, optimizing performance, and utilizing various design tools and techniques, the game's visual and interactive elements are carefully crafted to provide players with an immersive and visually appealing experience. The attention to detail

in the room design, the optimization of visuals, and the seamless integration of interactive features all contribute to creating a captivating and engaging gameplay environment.

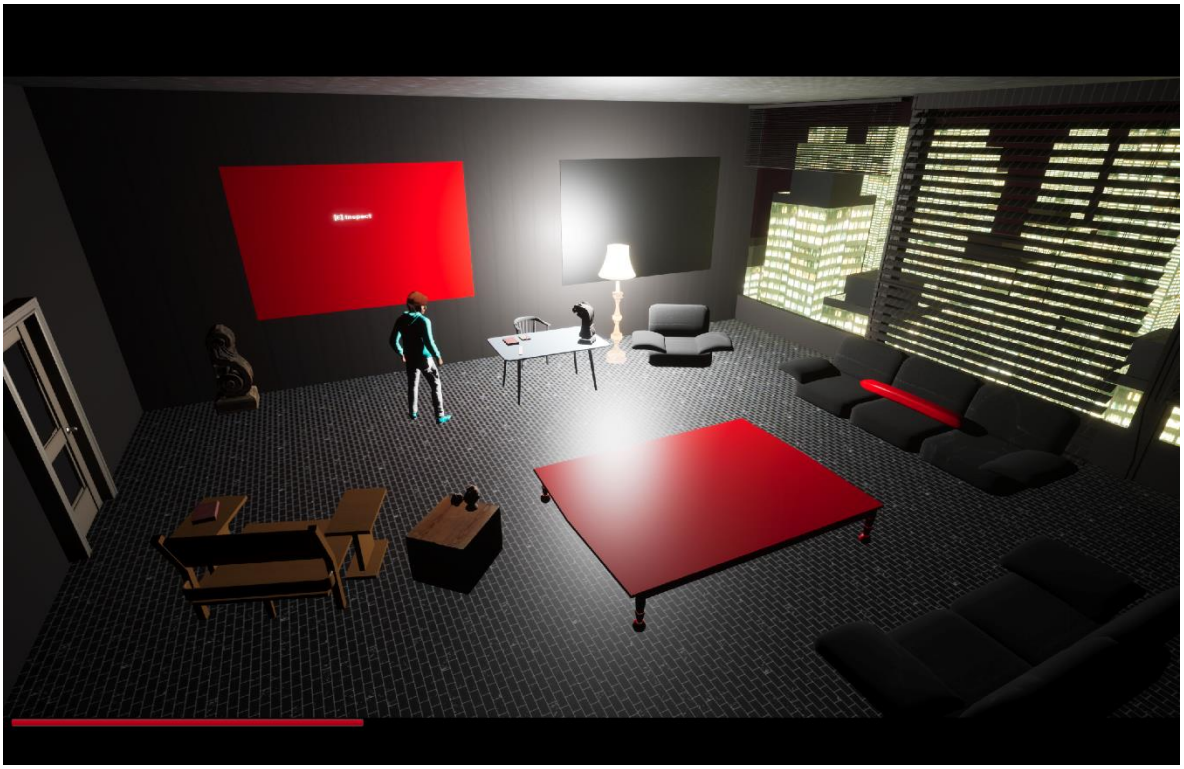


Image 4.4

4.5 CREATION OF LEVEL 3

In level three, extensive research was conducted on ancient Egypt and the Giza Pyramid. The research encompassed not only the interior of the pyramid but also scientific theories surrounding it which is about sound waves. According to our findings, the pyramid contains three chambers: one for the king, one for the queen, and one for hieroglyphs. However, the secrets within these chambers remain undiscovered. Non-playable characters (NPCs) were strategically placed within these chambers, and players must engage with them through solving small riddles to obtain information. The game presents a riddle based on the six deities, which serves as the encrypted code. Players are required to research the characteristics of these six deities and subsequently decipher the code using poetic clues. At

the conclusion of this puzzle, players will face a test on Anubis' scale, a deity associated with judgment and balance.

Additionally, the map features mysterious locations that can be accessed through specific paths, each harboring distinct challenges. There are two possible paths to reach the end of the game: the first relies on the player's parkour skills and the ability to hang onto ledges, while the second involves finding a hidden key with mystical properties. The first path tests the character's agility and platforming abilities, while the second path provides easier progression for those focused on exploration and research.

Moving on to the design and implementation aspects, the map was entirely designed by our team. In our two-dimensional thematic representation of the Egyptian pyramids, we utilized Blender to create several sculptures. The Egypt-themed assets from the Unreal Engine's free content library were used to populate our scenes, covering the objects we created. For the ground sections, we employed a cave theme, while rocky formations were transformed into blocks within the interior of the pyramid. Falling platforms were also implemented, triggering the creation of a block when the character steps on them. After a 5-second delay, gravity is activated on this static object, simulating its descent.

The design and implementation process prioritized optimization and performance. We utilized various techniques within the Unreal Engine to streamline gameplay mechanics and enhance visual effects. Additionally, sound design played a crucial role in immersing players in the game's atmosphere. Carefully selected music, sound effects, and dialogues complemented the overall experience, particularly within the mysterious and mystical ambiance of the pyramid's interior.

Overall, the combination of thorough research, immersive level design, engaging puzzles, and optimized implementation contributes to an enriching and captivating gaming experience in Level three.

Image 4.5

4.6 CREATION OF LEVEL 4 AND 5

In the final stages of the game, the character embarks on a journey to Brazil, armed with crucial clues and a razor-sharp enchanted dagger. This section of the game requires the character to employ the dagger's unique abilities to overcome specialized enemy soldiers. However, before infiltrating the fortress, the character must find a way inside. The design of the Brazilian environment is approached in two phases. In the first phase, the lush rainforests of Brazil are chosen, providing players with a fully immersive and expansive environment that encourages freedom of exploration. Throughout the rainforest, strategically placed materials facilitate the acquisition of valuable information. Furthermore, the design includes the incorporation of a customized vehicle, which is meticulously adjusted to navigate the challenging terrain. An intriguing design element is introduced by utilizing a green fabric

texture as a permeable surface, representing toxic water. This design choice not only enhances the narrative but also optimizes gameplay by providing a means to avoid the water. Upon entering the water, a regdoll animation is triggered to simulate the character's demise. To optimize performance, shadow quality for trees is reduced, and rocks of varying sizes are created using Blender and distributed based on their respective dimensions.

Within the fortress, the character encounters a statue puzzle that must be solved to progress. To create the puzzle, a poetic riddle is crafted, and the statues are redesigned using Blender, accompanied by material modifications. Each room within the fortress is meticulously designed, incorporating non-player characters (NPCs) to enhance the puzzle-solving experience. Upon solving the statue puzzle, the entrance to a gravity-free ventilation section is unlocked, employing a valve puzzle mechanism seen earlier in the game. This design choice allows the character to progress through a non-lethal route. Alternatively, another path exists, where the character must navigate through an area inhabited by highly advanced AI robots capable of immediately detecting and relentlessly attacking the character. This path presents a challenging and intense combat scenario. To provide hiding spots during these encounters, individual sandbags are created using Blender and stacked atop one another. A boss battle featuring a sniper enemy is designed, and an appropriate upward ladder is implemented to facilitate the "assassinate" mechanic.

In conclusion, the Brazilian chapter of the game offers a captivating and immersive experience, incorporating intricate puzzles, diverse environmental challenges, and intense combat scenarios.

Image 4.6

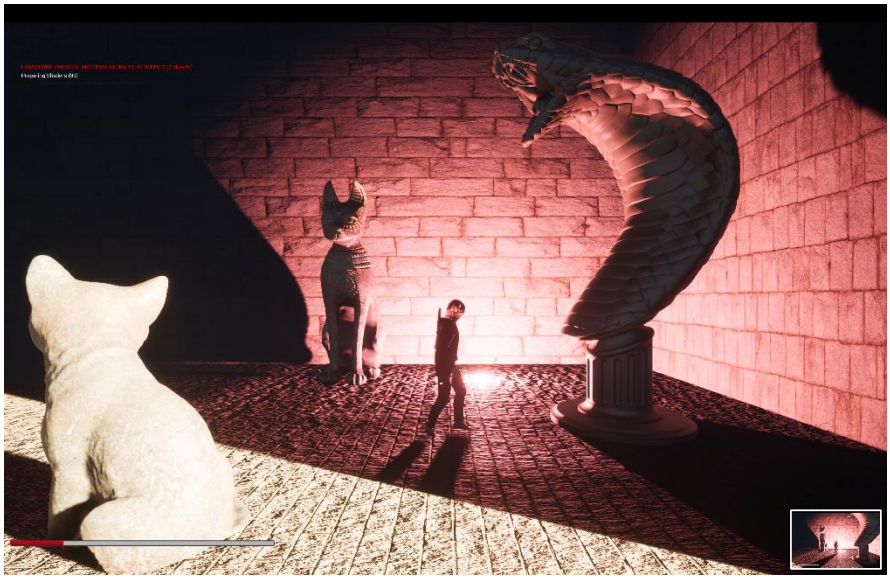


Image 4.7

4.7 MENU DESIGN

We used our school colors as a homage for the menu design. We designed the buttons and background using Adobe Illustrator and Photoshop. We called the texture quality, shadow quality, and pixel count with functions so that the volume can be adjusted.

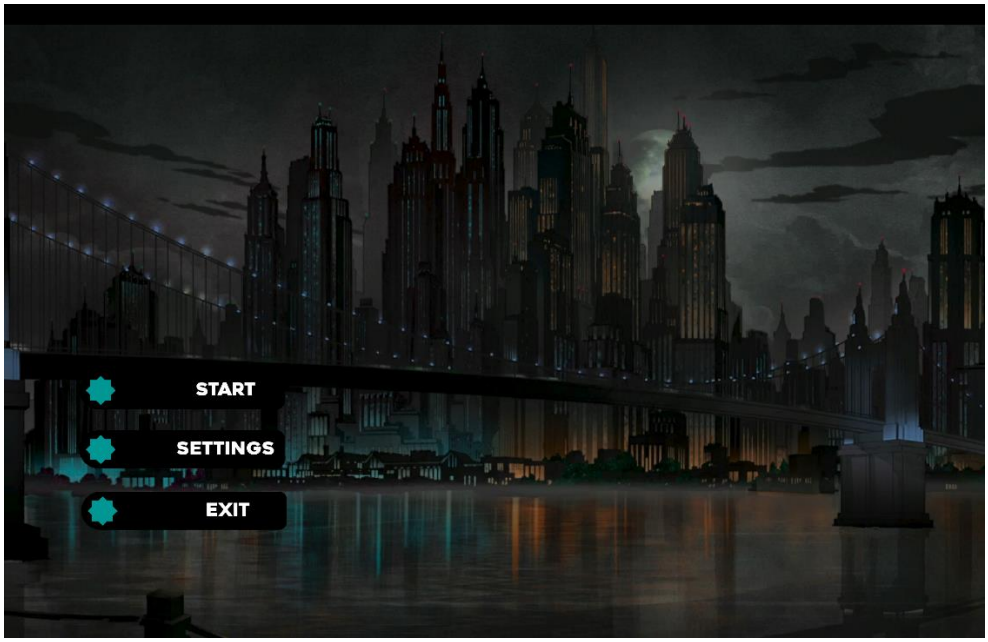


Image 4.8

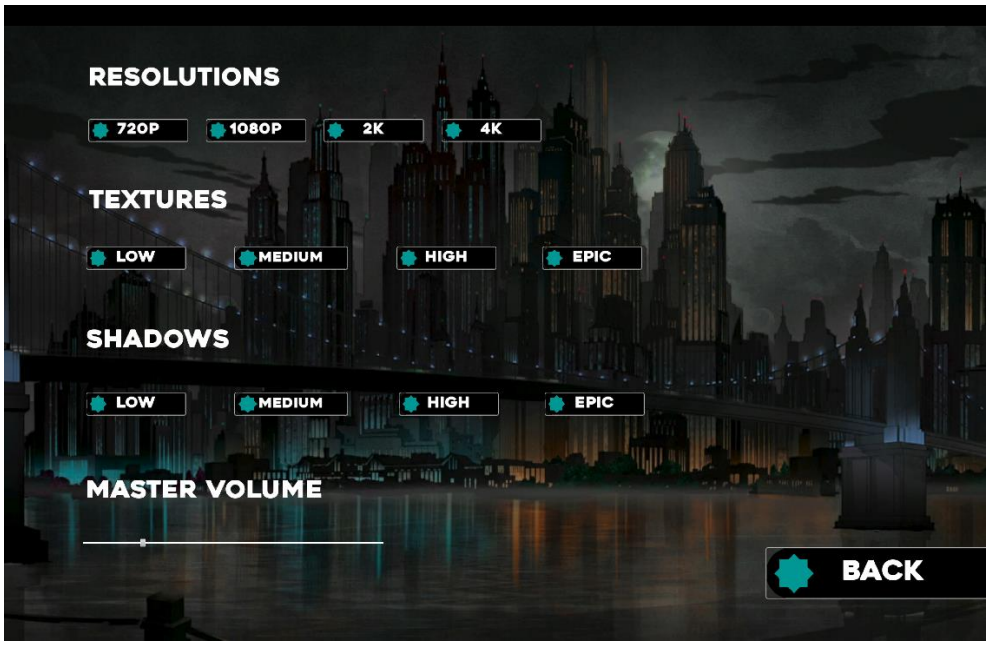


Image 4.9

5. INTERACTED WEBSITE

In today's digital era, technology continues to revolutionize the way we experience and interact with various mediums. One such innovation is the creation of interactive game websites that blend the excitement of gaming with the convenience of online platforms. This part explains a captivating website that combines React, QR codes, and other cutting-edge technologies to provide an immersive and engaging storytelling experience.

At the heart of this website lies an intriguing game that encourages players to embark on an exciting journey filled with mysteries and discoveries. with React, the website employs a variety of features, such as React Router, useEffect, React Reference, and local storage, to enhance the gameplay and ensure a flawless user experience. Through the careful placement of these QR codes, the website fosters an engaging sense of exploration and discovery. Players are motivated to search for QR codes within the game environment, stimulating their curiosity and further immersing them in the narrative. This interactive element not only

enhances the gameplay experience but also encourages players to actively engage with the website, creating a captivating and dynamic storytelling medium.

5.1 QR CODE IMPLEMENTATION

One of the distinctive features of this interactive game website is the incorporation of QR codes. These codes act as gateways to additional hints and character cards, enriching the narrative and offering players deeper insights into the game's world. By scanning the QR codes with their mobile devices, players are directly redirected to the website, where they gain access to relevant information that enhances their understanding of the story and aids their progress, you can see in image 5.1 and image 5.2 first and second QR code of game it will redirect you to our website image 5.1 redirects to first character card and image 5.2 redirects you to first hint. In image 5.3 you can see the website without any hint or character codes given, after we given a character code by scanning QR code or add code manually you could see that we can access character in the site. If player add an important hint from game a storyline will appear in website. You can see the website after adding a character and a hint in image 5.4 and you can see local storage of that browser in image 5.5 All the QR codes inside the game are created in “https://www.qrcode-monkey.com”.

Image 5.1

Image 5.2

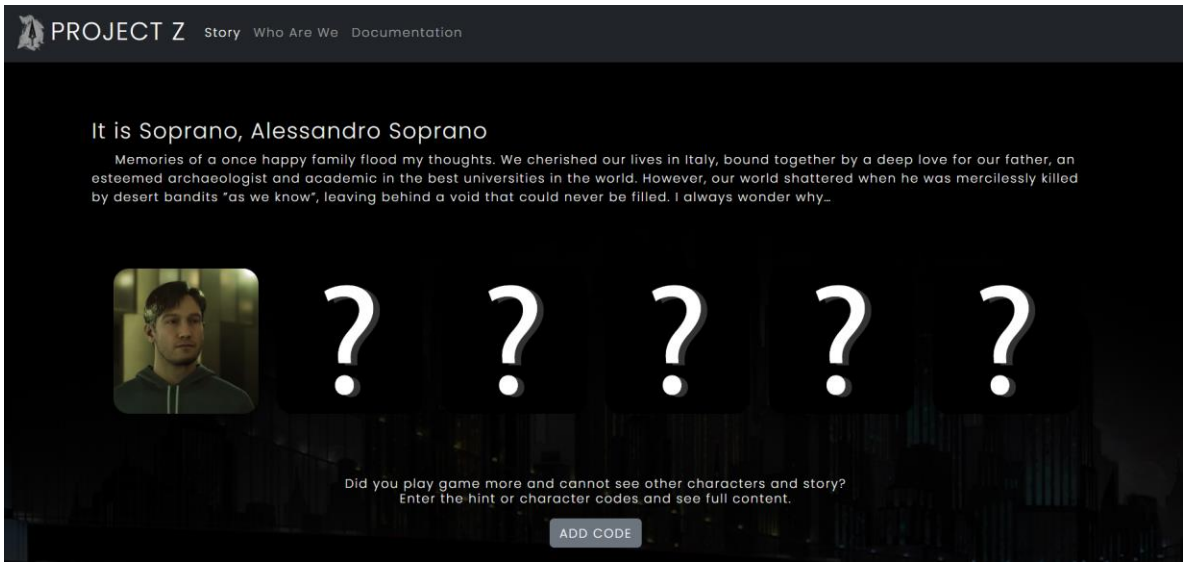


Image 5.3

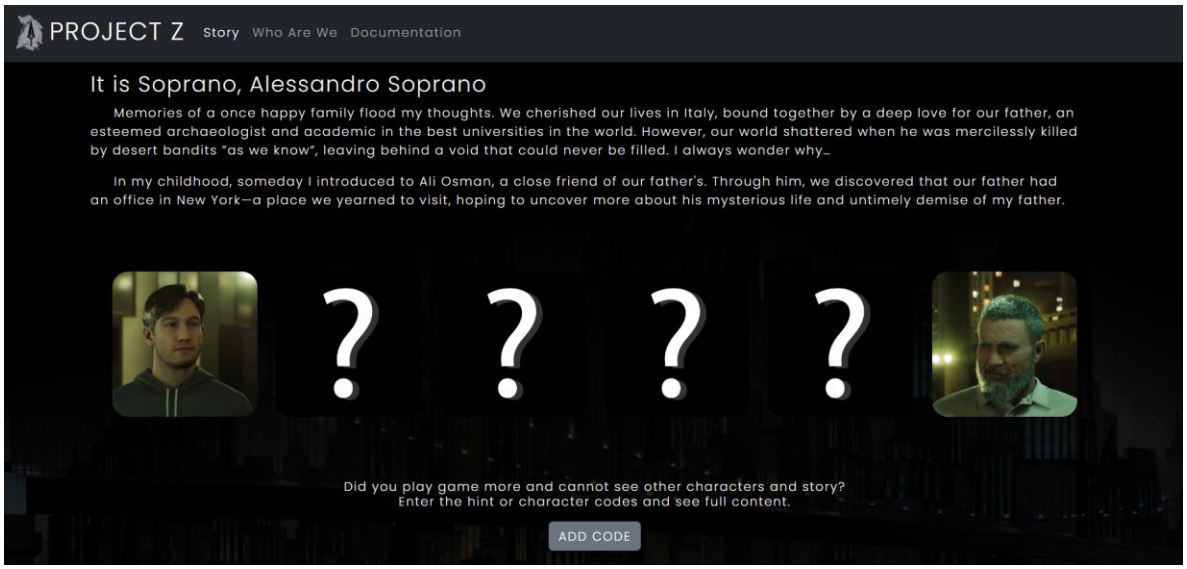


Image 5.4

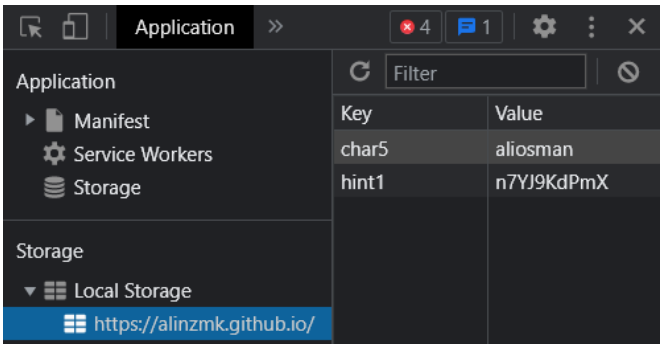
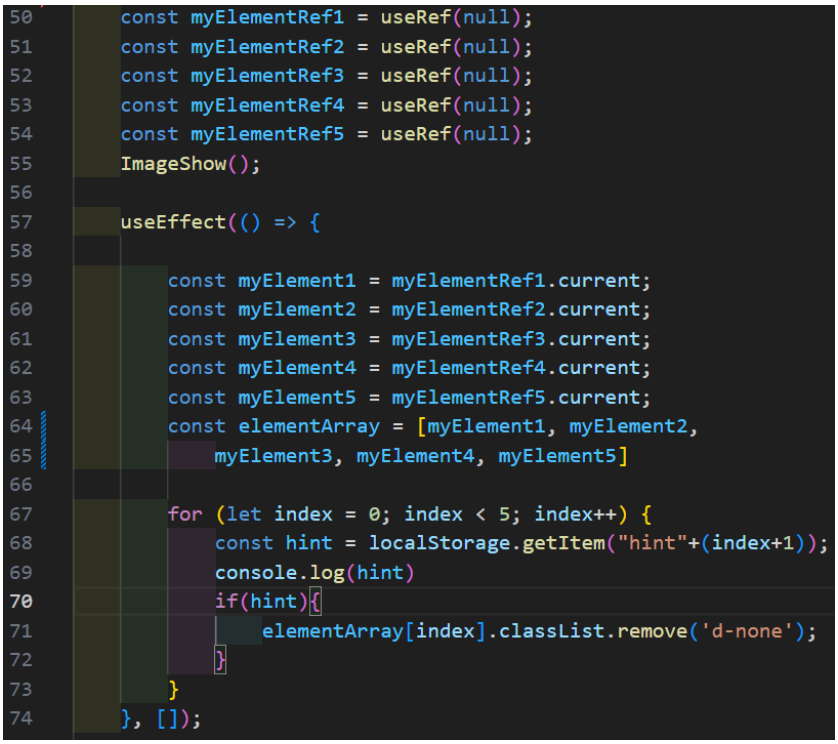


Image 5.5

In this project we use React Router for navigation because of React Router allows smooth navigation between different game sections, providing players with a clear and intuitive interface. Through the clever utilization of useEffect, the website can trigger story parts and character cards within taking input from local storage. React Reference serves as a valuable tool to access and manipulate elements on the webpage, enabling players to interact with story of the game you can see the usage of useEffect and references in image 5.6 and image 5.7. Additionally, the integration of local storage ensures that players can save their progress and return to the game at their convenience, without losing any crucial information even if they change their browser players can add codes by hand with “Add Code” modal in image 5.8.



```
    const myElementRef1 = useRef(null);
    const myElementRef2 = useRef(null);
    const myElementRef3 = useRef(null);
    const myElementRef4 = useRef(null);
    const myElementRef5 = useRef(null);
    ImageShow();

    useEffect(() => {

        const myElement1 = myElementRef1.current;
        const myElement2 = myElementRef2.current;
        const myElement3 = myElementRef3.current;
        const myElement4 = myElementRef4.current;
        const myElement5 = myElementRef5.current;
        const elementArray = [myElement1, myElement2,
            myElement3, myElement4, myElement5]

        for (let index = 0; index < 5; index++) {
            const hint = localStorage.getItem("hint"+(index+1));
            console.log(hint)
            if(hint){
                elementArray[index].classList.remove('d-none');
            }
        }
    }, []);
```

Image 5.6

```
function ImageShow(){
    var paragArray = [];
    for(let i=0;i<5;i++){
        const charCode = ["ettore", "abraham", "levanja", "dhruv", "aliosman"];
        const x = localStorage.getItem("char"+(i+1));
        if(charCode[i] === x){
            if(i==0){
                p2 = michaelPhoto; c2 = "/Ettore-M5sCp";
            }
            else if(i==1){
                p3 = anastasiosPhoto; c3 = "/Abraham-b7HkP";
            }
            else if(i==2){
                p4 = laraPhoto; c4 = "/Levanja-G4tE9";
            }
            else if(i==4){
                p5 = aliosmanPhoto; c5 = "/AliOsman-2nL1f";
            }
            else if(i==3){
                p6 = dhruvPhoto; c6 = "/Dhruv-3t5aq";
            }
        }
    }
    return paragArray
}
```

Image 5.7

Image 5.8

5.2 ACCESSIBILITY AND PUBLISHMENT:

One notable aspect of this interactive game website is its accessibility. By utilizing GitHub Pages, the website can be published freely, allowing a wider audience to experience the thrill of the game without any financial barriers. This democratization of access ensures that players from various backgrounds can enjoy the storytelling with the innovative gameplay mechanics. Site is published on https://alinzmk.github.io/ProjectZ/

In conclusion, the interactive game website discussed in this part represents a remarkable fusion of technology and adventure. Leveraging the power of React and incorporating QR codes, the website provides players with an engaging and immersive storytelling experience. Through features like React Router, useEffect, React Reference, and local storage, the gameplay is enhanced, and the user experience is optimized. Moreover, the website's accessibility through GitHub Pages ensures that this enthralling adventure can be enjoyed by a diverse audience. As technology continues to advance, such interactive game websites hold immense potential in revolutionizing the way we engage with narratives, combining the best of both digital innovation and storytelling traditions.

6. DISCUSSION AND RESULTS

The development and integration of QR codes within the game, utilizing Unreal Engine 5, proved to be a successful endeavor, providing players with an interactive and engaging experience. The following discussion presents the results and outcomes of the project.

During gameplay, players were able to scan QR codes, which led them to the dedicated website providing specific hints and guidance. This dynamic hint system proved effective in assisting players in overcoming challenges and progressing through the game. The tailored hints provided valuable information without giving away complete solutions, allowing players to exercise their problem-solving skills and maintain a sense of accomplishment.

The use of QR codes as interactive elements within the game added an extra layer of immersion and interactivity. Players were encouraged to explore their surroundings, scan QR codes, and decipher the provided clues. This aspect enhanced the overall gameplay experience, making it more interactive and rewarding for players.

The dedicated website associated with the QR codes played a crucial role in providing comprehensive information and hints. It served as a centralized resource where players could refer to for additional support whenever needed. The website's user-friendly interface and organized presentation of hints contributed to a smooth and seamless user experience.

Feedback from players indicated a positive reception towards the QR code integration and the dynamic hint system. Players appreciated the additional challenge and the opportunity to access hints tailored to each encountered obstacle. The incorporation of QR codes and the associated website added depth to the game and encouraged players to think critically and strategically.

The integration of Unreal Engine 5 proved beneficial in creating a visually stunning and immersive virtual environment. The advanced graphical capabilities and

design tools offered by the engine helped in crafting a captivating game world that enhanced players' overall experience.

Overall, the results of this project demonstrate the successful integration of QR codes and a dynamic hint system within a game developed using Unreal Engine 5. The QR codes added interactivity and immersion, while the dedicated website provided valuable hints and guidance. This combination resulted in an engaging and rewarding gameplay experience for players, encouraging critical thinking and problem-solving skills.

Future iterations of this project could explore further enhancements to the hint system, such as adaptive hints that adjust based on player progress, or the integration of QR codes with real-world objects to bridge the gap between virtual and physical experiences. Additionally, gathering more extensive feedback and conducting user studies could provide deeper insights into the impact of the QR code-driven gameplay on player engagement and satisfaction.

In conclusion, the successful implementation of QR codes and the dynamic hint system in this game showcases the potential for integrating novel technologies into gaming experiences. This project contributes to the growing body of knowledge in the field of interactive game design and highlights the effectiveness of QR codes as a tool for enhancing gameplay and player engagement.

BIBLIOGRAPHY

Science.org., 2017. Cosmic rays reveal unknown void in the Great Pyramid of Giza
< https://www.science.org/content/article/cosmic-rays-reveal-unknown-void-great-pyramid-giza >

Nature.com., 2017 Cosmic-ray particles reveal secret chamber in Egypt's Great Pyramid
< https://www.nature.com/articles/nature.2017.22939>

Barış Özcan Youtube Channel., 2022. Büyük Piramidin içinde ne var? 3B TUR
< https://youtu.be/GW3MMtJMeq4>

React Documentation., 2023. React Official Documentation
<https://react.dev/blog/2023/03/16/introducing-react-dev>

PerdoTech Youtube Channel., 2022. How To Deploy A React App To Github Pages <https://youtu.be/Q9n2mLqXFpU>

React Router Documentation., 2021. React Router Official Documentation
< https://reactrouter.com/en/main>

Gorka Games Youtube Channel., 2023. Unreal Engine 5 RPG Tutorial Series
<https://www.youtube.com/watch?v=FNTyIWkv5k8&list=PLiSlOaRBfgkcPAhYpGps16PT_9f28amXi&pp=iAQB >

Matt Aspland Youtube Channel., 2021. Third Person Shooter Minigame Unreal Engine
<https://www.youtube.com/watch?v=95oz8V5U8GQ&list=PLQN3U_-lMANNDOnDEtIWQ5GaBtzbTI1Sf&pp=iAQB >

Reids Channel, Youtube Channel., 2020. Unreal Engine 4 Dialog System Tutorial
< https://youtu.be/xI7WHD5C-ps >

CURRICULIM VITAE-1

Name Surname	: Ali Nazım KÜÇÜK
Place of Birth	: TRABZON
Date of Birth	: 31.12.2000
Marital Status	: Single
Foreign Language	: English

Educational Background (Institution and Year)

High School : Trabzon Final Fundamental High School (2018)

Undergraduate : Alanya Alaaddin Keykubat University Rafet Kayış Faculty of Engineering Computer Engineering (2023)

Worked/Working Institutions and Year

Trabzon Metropolitan Municipality / Intern (2022)

Phi Software / Intern (2022)

CURRICULIM VITAE-2

Name Surname	: Ümit Numan DUMAN
Place of Birth	: SAMSUN
Date of Birth	: 04.08.2000
Marital Status	: Single
Foreign Language	: English

Educational Background (Institution and Year)

High School : Ordu Ünye Final Fundamental High School (2018)

Undergraduate : Alanya Alaaddin Keykubat University Rafet Kayış Faculty of Engineering Computer Engineering (2024)

CURRICULIM VITAE-3

Name Surname	: Halil İbrahim ÖZDEMİR
Place of Birth	: Ünye/ORDU
Date of Birth	: 06.10.2000
Marital Status	: Single
Foreign Language	: English

Educational Background (Institution and Year)

High School : Ankara Açı Fundamental High School (2018)

Undergraduate : Alanya Alaaddin Keykubat University Rafet Kayış Faculty of Engineering Computer Engineering (2023)

Worked/Working Institutions and Year

Roketsan /Intern (2022)