



A SYNOPSIS ON
“Erica-The Warrior”

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

To

Guru Gobind Singh Indraprastha University, Delhi

Under the Guidance of:

Dr. Barkha Bahl

Director

Submitted By:

Ankur Chaurasia (35120602019)

BCA 6th Semester



TRINITY INSTITUTE OF PROFESSIONAL STUDIES

(Affiliated to Guru Gobind Singh Indraprastha University, Delhi)

Ranked “A+” Institution by SFRC, Govt. of NCT of India

Recognized under section 2(f) of the UGC Act, 1956

NAAC Accredited “B++” Grade Institution



CONTENTS

S. NO.	Topic	Page No.
1	Introduction	3
2	Motivation	7
3	Methodology	8
4	Tools: Software & Hardware Requirement	12
5	System Testing	13
6	Future Scope	14
7	Conclusion	15
8	References	16



INTRODUCTION

Nowadays, it is a truth that video games are the world's most beloved merriment industry. Even though video games are frequently played by kids and youth, there are many older people who also play those games. Moreover, video games are progressively becoming even more of a part of our culture. Today's kids and human beings are wasting their time on technological devices by playing video games. This examination is planned as an entrance to present thought as regards turn-based RPG strategy video games' role. Role-Playing Games (RPGs) form one of the primary genres of games and are possibly one of the most widely varied game forms around. Furthermore, RPGs have been designed for or ported to every existing hardware platform produced for games, in addition to tabletop and live-action variations. RPGs formed one of the primary sources of inspiration for the early evolution of computer games and retain a strong influence to this date.

The game engine ensures game originators withal the obligatory characteristics to construct games rapidly, influentially, and yieldingly. Game engines are too momentous for developing video games and muster numerous primaries structs. It is kind of a spine for game development.

For a game developer, this part is the most crucial decision. Game engine choice is able to repulse the standard of the game advanced or back in whichever facet. The most important familiar game engines own their own ways and characteristics. There are any inquiries and responses your necessity to do your selection any of them the size of your squad, your warrant on the game engine, the structure of your project, your license the estimates, your choice of many details, such as 2D, 3D, mobile, console, and PC. For our game we have chosen Unreal Engine.



Unreal Engine 5



Unreal Engine 5 (UE5) is now available as an Early Access build. We are excited to share this new version of Unreal with the world, giving everybody a chance to get hands-on with the future of Unreal Engine.

WORLD BUILDING AND COLLABORATION

One of our ongoing goals is to make the creation of dynamic, open worlds faster, easier, and more collaborative for teams of all sizes. With Unreal Engine 5, a new World Partition system changes how levels are managed and streamed, automatically dividing the world into a grid and streaming the necessary cells. Additionally, teams can simultaneously work in the same World without treading on each other's toes, thanks to the new One File Per Actor system, which breaks each level into a series of files. This means that when you make changes to the level, you are only saving and submitting changes to the individual Actors you've changed or added. There have also been significant improvements to how Unreal works with Source Control,



specifically with Perforce, allowing developers to do almost all of their SCM operations directly in the editor.

NEW USER INTERFACE

In Unreal Engine 5, the Unreal Editor gets a makeover, with an updated visual style, streamlined workflows, and optimized use of screen real estate, making it easier, faster, and more pleasing to use. To free up more space for viewport interactions, we've added the ability to easily summon and stow the Content Browser and to dock any editor tab to the collapsible sidebar. You can now quickly access frequently used properties in the Details panel with a new favouriting system, while the new Create button on the main toolbar lets you easily place Actors into your world. There's also a streamlined, easier workflow for creating new projects.

CHAOS PHYSICS

Chaos Physics is a lightweight physics simulation solution available in Unreal Engine 5, built from the ground up to meet the needs of next-generation games. The system includes the following major features:

- Rigid Body Dynamics
- Rigid Body Animation Nodes and Cloth Physics
- Destruction
- Ragdoll Physics
- Vehicles
- Physics Fields
- Fluid Simulation
- Hair Simulation



MIXAMO

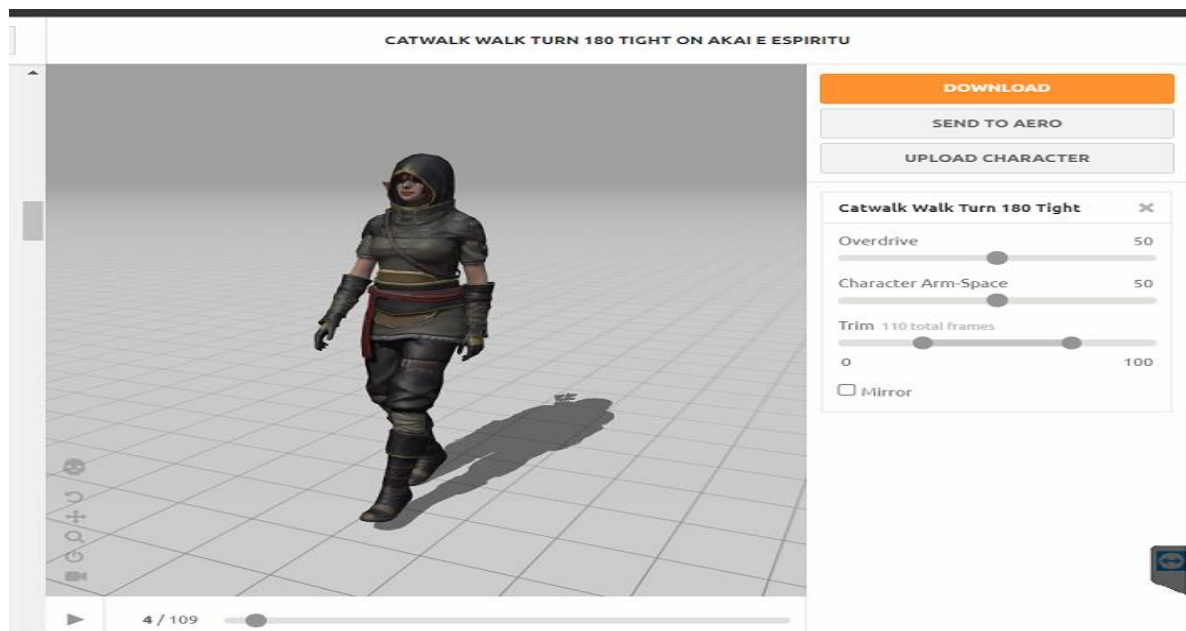


mixamo

Mixamo is a 3D computer graphics technology company. Based in San Francisco, the company develops and sells web-based services for 3D character animation. Mixamo's technologies use machine learning methods to automate the steps of the character animation process, including 3D modeling to rigging and 3D animation.

Edit the animation

1. Use the editor panel to control the animation settings such as speed and arm spacing after you've chosen an animation and applied it to your character.



You can remove an animation by clicking the **X** in the top of the animation editing section to return to your static character.

2. Select the **Download** button when you're finished editing the animation.



Motivation

Our aim for this project is to make a turn-based RPG game. We intend to make it run on PCs and laptops. We used the Unreal Engine 5 to develop our game. We wanted to develop a role-playing video game within the electronic game genre in which players advance through a story quest and often many side quests, for which their character or party of characters gain experience that improves various attributes and abilities.

This is not something we would develop for the sake of a minor project, but something that is truly of our interest, will help us better understand gaming technology, and is also very different and unique. We did a little research about which engine should be the best platform for developing our game, and the Unreal Engine came out as the best of all.

We did face a lot of issues and practical problems, but we overcame them all with the guidance of our teachers and research. The most important thing about our project is that the future scope is endless and every new idea is welcomed.



METHODOLOGY

Unreal Engine enables game developers and creators across industries to realize next-generation real-time 3D content and experiences with greater freedom, fidelity, and flexibility than ever before.

Unreal Engine 5 gets a UI makeover!

With Quixel Bridge now fully integrated, you have drag-and-drop access to the entire Megascans library. It's part of the new Create menu for acquiring content and creating and placing Actors.

To free up space in the viewport, you can now easily summon and stow the Content Browser and dock any editor tab to a sidebar. Plus, it's faster to find the properties you're looking for in the Details panel.

The **Blueprint Visual Scripting** system in Unreal Engine is a complete gameplay scripting system based on the concept of using a node-based interface to create gameplay elements from within Unreal Editor. As with many common scripting languages, it is used to define object-oriented (OO) classes or objects in the engine. As you use UE4, you'll often find that objects defined using Blueprint are colloquially referred to as just "Blueprints."

This system is extremely flexible and powerful as it provides the ability for designers to use virtually the full range of concepts and tools generally only available to programmers. In addition, Blueprint-specific markup available in Unreal Engine's C++ implementation enables programmers to create baseline systems that can be extended by designers.

How Do Blueprints Work?

In their basic form, Blueprints are visually scripted additions to your game. By connecting Nodes, Events, Functions, and Variables with Wires, it is possible to create complex gameplay elements.

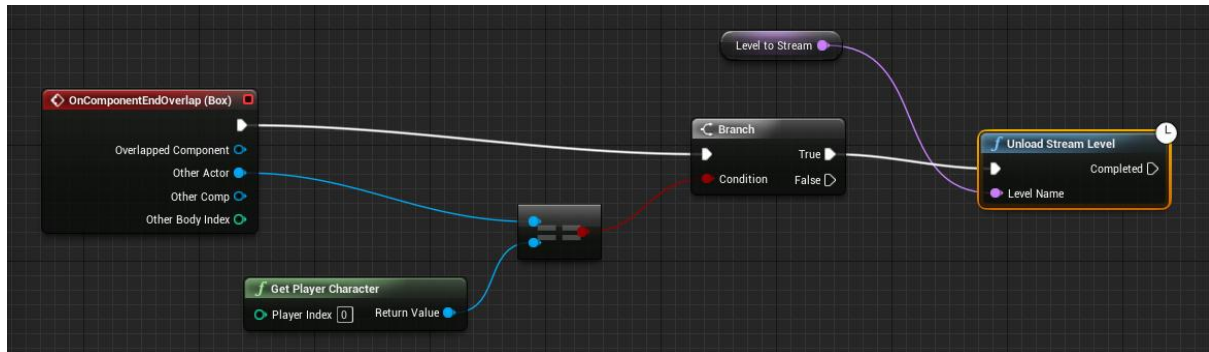
Blueprints work by using graphs of Nodes for various purposes - object construction, individual functions, and general gameplay events - that are specific to each instance of the Blueprint in order to implement behavior and other functionality.

Commonly Used Blueprint Types

The most common Blueprint types you will be working with are **Level Blueprints** and **Blueprint Classes**.

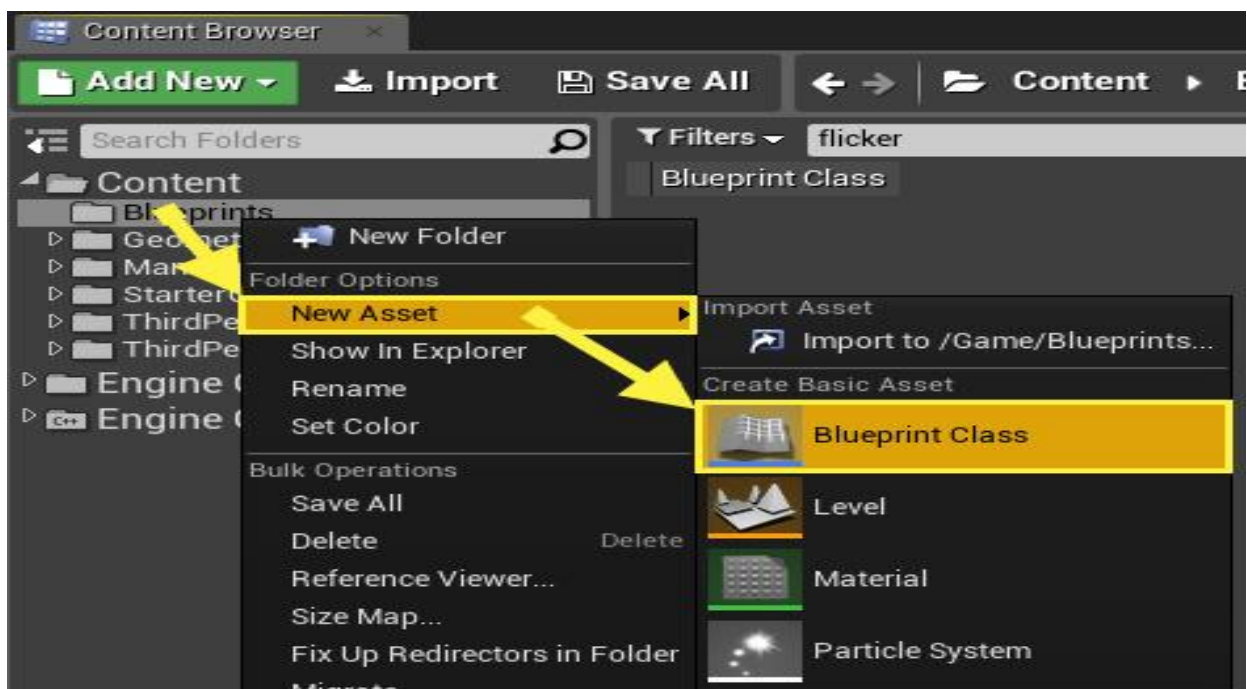


Level Blueprint



The Level Blueprint fills the same role that Kismet did in Unreal Engine 3, and has the same capabilities. Each level has its own Level Blueprint, and this can reference and manipulate Actors within the level, control cinematics using Matinee Actors, and manage things like level streaming, checkpoints, and other level-related systems. The Level Blueprint can also interact with Blueprint Classes (see the next section for examples of these) placed in the level, such as reading/setting any variables or triggering custom events they might contain.

Blueprint Class



Blueprint Classes are ideal for making interactive assets such as doors, switches, collectible items, and destructible scenery. In the image above, the button and the set of doors are each separate Blueprints that contain the necessary script to respond to player overlap events, make them animate, play sound effects, and change their materials (the button lights up when pressed, for example).



In this case, pressing the button activates an event inside the door Blueprint, causing it to open - but the doors could just as easily be activated by another type of Blueprint, or by a Level Blueprint sequence. Because of the self-contained nature of Blueprints, they can be constructed in such a way that you can drop them into a level and they will simply work, with minimal setup required. This also means that editing a Blueprint that is in use throughout a project will update every instance of it.

What Else Can Blueprints Do?

You have read about Level Blueprints and Blueprint Classes, listed below are a handful of examples that can be accomplished with the Blueprint system.

Create Customizable Prefabs with Construction Scripts

The Construction Script is a type of graph within Blueprint Classes that executes when that Actor is placed or updated in the editor, but not during gameplay. It is useful for creating easily customizable props that allow environment artists to work faster, such as a light fixture that automatically updates its material to match the color and brightness of its point light component, or a Blueprint that randomly scatters foliage meshes over an area.

In the Content Examples maps, the long rooms that contain each example (pictured above) are actually a single Blueprint made up of many components. The Blueprint's Construction Script creates and arranges the various Static Meshes and lights according to parameters exposed in the Blueprint's **Details** panel. With each Content Example map we created, we were able to drop in the demo room Blueprint, set values for the length, height, and number of rooms that would be generated (and a few other options), and have a complete set of rooms ready in moments.

A Blueprint like this can be time-consuming to create initially, but if you know you will use it often, the time saved when building a level and the ease of making changes can make it very worthwhile.

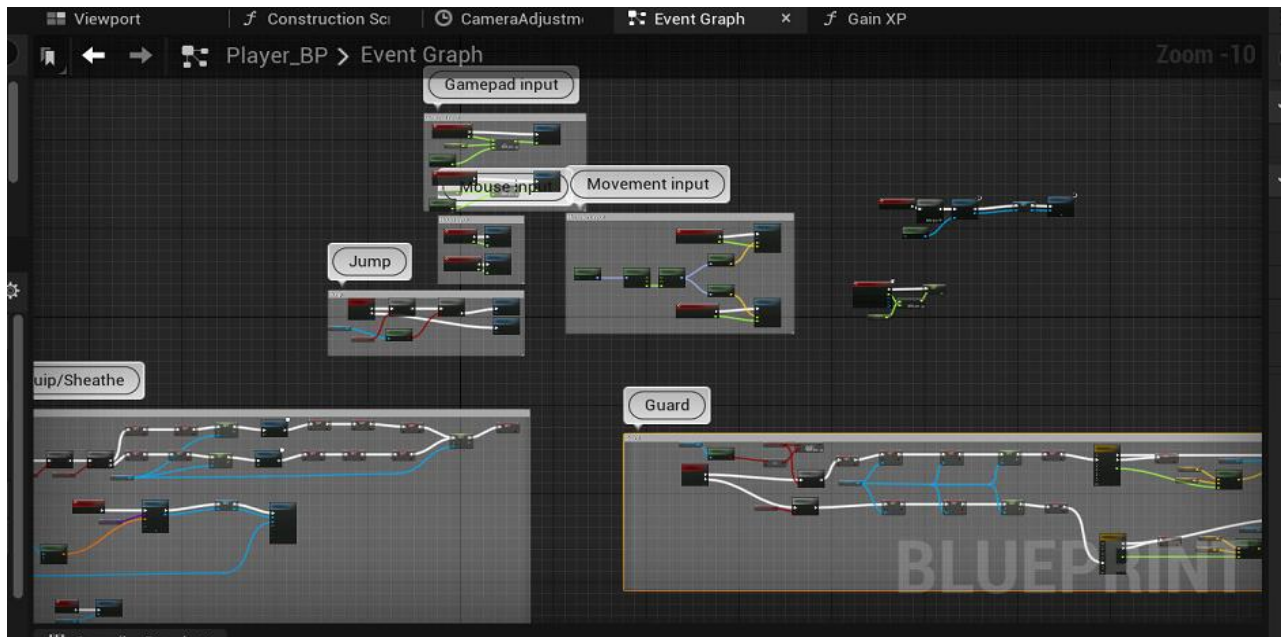
Create A Playable Game Character

Pawns are also a type of Blueprint Class, and it is possible to put together every element you need for a playable character in the Blueprint graph. You can manipulate camera behavior, set up input events for mouse, controller, and touch screens, and create an Animation Blueprint asset for handling skeletal mesh animations.



When you create a new Character Blueprint, it comes with a character component that has much of the behavior needed for moving around, jumping, swimming, and falling built-in, and all that is required is to add some input events in accordance with how you want your character to be controlled.

Create A HUD



Blueprint script can be used to create a game's HUD as well, which is similar to Blueprint Classes in that it can contain event sequences and variables, but is assigned to your project's GameMode asset instead of being added directly to a level.

You can set up a HUD to read variables from other Blueprints and use them to display a health bar, update a score value, display objective markers, and so on. It is also possible to use the HUD to add hit-boxes for elements like buttons that can be clicked on or, in the case of mobile games, can respond to touch input.

Modules: -

1. Engine download and project setup.
2. Character Initialization.
3. Equipping Weapons
4. Setting up attack and Damage Variables
5. Developing an AI based enemy.



TOOLS: SOFTWARE & HARDWARE REQUIREMENTS

Hardware Requirements: -

- CPU: 2.9GHz Intel Core i7-7500U (dual-core, 4MB cache, up to 3.5GHz)
- RAM: 8 GB DDR4
- Graphics: Intel HD Graphics 620
- Storage: 1 TB Hard Drive
- Keyboard & Mouse
- Active Internet Connection

Software Requirements: -

- Unreal Engine 5
- Mixamo



SYSTEM TESTING

- **Testing:** Testing is a major part of creating our website as we want our customer use a problem free sign recognition website. For this, we test the site over multiple browsers and devices to check whether it is working like supposed to. As at this point, the URL is private and should launch after confirming all the things.
- **Unit Testing:** It focuses verification effort on the smallest unit of software design, the module.
- **Conditional Testing:** In this part of the testing, each of the conditions were tested to both true and false aspects. And all the resulting paths were tested so that each path that may generate on particular condition is traced to uncover any possible errors.
- **Data Flow Testing:** This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The definition-use chain method was used in this type of testing. These were particularly useful in nested statements.
- **Loop Testing:** In this type of testing, all the loops are tested to all the possible limits. All the loops were tested at their limits, just above them and just below them. For nested loops test, the inner most loop first and then work outwards. For concatenated loops, the values of dependent loops were set with the help of connected loop. Unstructured loops were resolved into nested loops or concatenated loops are tested as above.



FUTURE SCOPE

- This game is made for PC in future we can make an IOS version & Android version.
- We will add a database to our game which will store the process of the player.
- We can add multiple characters to our game.
- We can also create dynamic environments and levels to make the experience more interactive for the users.
- We can also create a setting for customizing graphics for end users.
- We can also add cut-scenes to add a story line to the game.
- We can also add the feature of multiplayer.






CONCLUSION

Erica-The Warrior can be the most fascinating role-playing game for the end users. Role-playing games sit at the intersection of four phenomena – play, roles, games, and media culture. Role-playing games are quite interactive.

Erica-The Warrior is free to move anywhere in the environment and can perform magic spells, hit, kill, and heal herself. This open world game also has many animals to make it look more lively, as well as many other graphical animations that can entice users to play our game. Unreal Engine fastens to layouts, colour, sound, physics, and animation. Today's contemporary game engines supply realistic. graphics. Game engine choice could repulse the standard of the game advanced or back in any facet.



REFERENCES

-  Unreal Engine 5: <https://www.unrealengine.com/en-US/?sessionInvalidated=true>
-  Mixamo: <https://www.mixamo.com/#/>
-  <https://www.slideshare.net/NadiaIIT/final-project-report-of-a-game>
-  https://www.researchgate.net/publication/330942093_Video_Game_Report_-_Riko_The_Adventurer
-  <http://norma.ncirl.ie/2390/1/kellymcguinness.pdf>
-  https://www.theseus.fi/bitstream/handle/10024/71525/Xia_Peng.pdf?sequence=1&isAllowed=y
-  <https://www.scribd.com/document/433077579/Project-Report-Game>
-  Cabbit. (2014). 24x32 Characters Big Pack. Available from: <https://opengameart.org/content/24x32-characters-with-faces-big-pack> [Accessed on 15 December 2018]
-  Beast. (2011). Sewer Tileset. Available from: <https://opengameart.org/content/sewer-tileset> [Accessed on 15 December 2018]
-  Angelee. (2018). Little Servant Devil Animation. Available from: <https://opengameart.org/content/little-servant-devil-animation> [Accessed on 15 December 2018]
-  C Tatz. (2014). Free UI Asset Pack 1. Available from: <https://opengameart.org/content/free-ui-asset-pack-1> [Accessed on 30 December 2018]
-  Kenney. (2014). Onscreen Controls (8 styles). Available from: <https://opengameart.org/content/onscreen-controls-8-styles> [Accessed on 30 December 2018]