Integrating AI with Meta Human Avatars in Unreal Engine

A complete hands-on guide to implementing smart NPCs in Unreal Engine using Convai.

ICVR 2025
International Conference on Virtual Reality

Overview: From Setup to Smart NPCs with LLMs

1. Introduction
   - What are Large Language Models (LLMs)?
   - Brief History of LLMs
   - Why integrate them into games and virtual worlds?

2. LLMs in Virtual Reality Games (based on research)
   - Dynamic NPC Interactions
   - Procedural Storytelling
   - Intelligent Game Masters
   - Personalized Player Experience
   - Accessibility, Inclusivity, and Usability

3. Useful Plugins in UE 5
   - Some related plugins and what they are used for
   - What is Convai?

4. Unreal Engine Setup
   - UE 5 and project initialization

5. Advanced Integration
   - Step-by-step Convai plugin integration

6. TUMSPhere

7. Q&A

What is a Large Language Model (LLM)?

• LLMs are neural network systems trained on massive text data to understand and generate human-like languages.

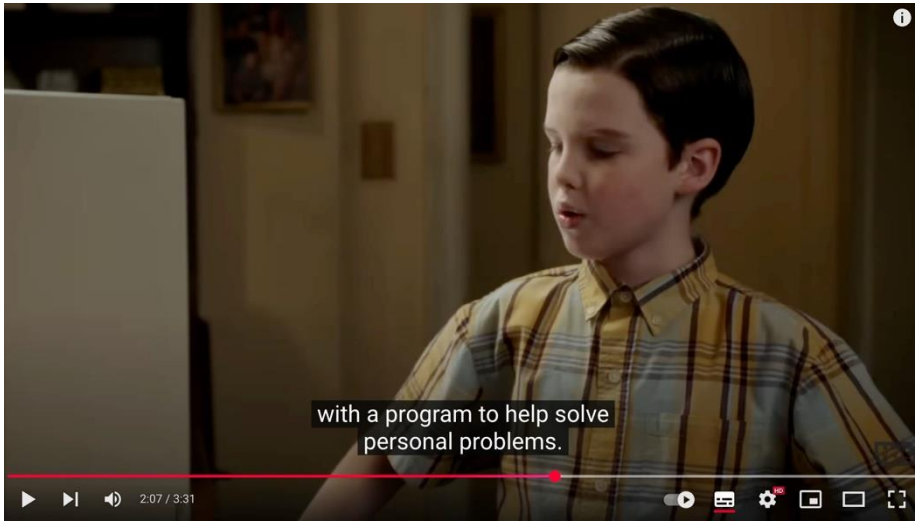
Recent LLM models you've probably heard of or even used:

- **GPT (OpenAI)** – powering ChatGPT
- **BERT (Google)** – known for deep text understanding (used in Google Search)
- **LLaMA (Meta)** – open-source, efficient
- **Gemini (Google DeepMind)** – multimodal model integrated into Google products
- **PaLM (Google)** – integrated into Bard (now named as Gemini)
- **Claude (Anthropic)** – known for long context and helpfulness

A Brief Story of LLMs

- **Pre-2010: Rule-Based & Statistical NLP**
  - ➢ Early systems relied on handcrafted rules and statistical models (e.g., n-grams).
  - ➢ Limited to domain-specific tasks, poor scalability.

Natural Language Processing (NLP) is the field of AI focused on enabling machines to **understand, interpret, and generate human language.** It is like the intersection of **linguistics, computer science, and machine learning**, powering applications like chatbots, translation, voice assistants, and LLMs.



A Brief Story of LLMs

•2013–2017: Word Embeddings & RNNs

- Word2Vec, GloVe introduced vector-based word meaning, they represented words in continuous vector spaces, capturing semantic relationships between words.
- Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) enabled more context-aware language processing and allowed for more improved text generation and sentiment analysis.

• 2017: The Transformer

- Vaswani et al.'s paper *"Attention Is All You Need"* introduced transformers.
- Parallel training + self-attention = breakthrough in scalability and context comprehension.

• 2018–2019: Pretrained Transformers (BERT, GPT-2)

- BERT used bidirectional context, excelling in understanding tasks.
- GPT-2 used autoregressive decoding, generating fluent, coherent text.

A Brief Story of LLMs

- **2020 - 2022: Scaling Up → LLM Era**
  - GPT-3 (175B params), GPT-4, Claude, PaLM, LLaMA, etc.
  - Emergent abilities like reasoning, coding, vision integration (e.g., GPT-4V).

- **2023–2024: The Rise of Multimodal and Optimized LLMs**
  - OpenAI released **GPT-4**, introducing stronger reasoning, safer responses, and extended context windows (up to 32K tokens).
  - Later in 2023, **GPT-4 Turbo** was launched — a cheaper, faster version optimized for ChatGPT and API use, becoming the default model in ChatGPT Pro.
  - In 2024, **GPT-4o ("omni")** brought native multimodality, combining **text, vision, and audio** in a single model — enabling real-time, speech-based, and visually grounded interactions.

Around the same time, **Claude 1** by Anthropic emerged as a leading alternative, praised for its human-aligned conversation style and robust reasoning.

A Brief Story of LLMs

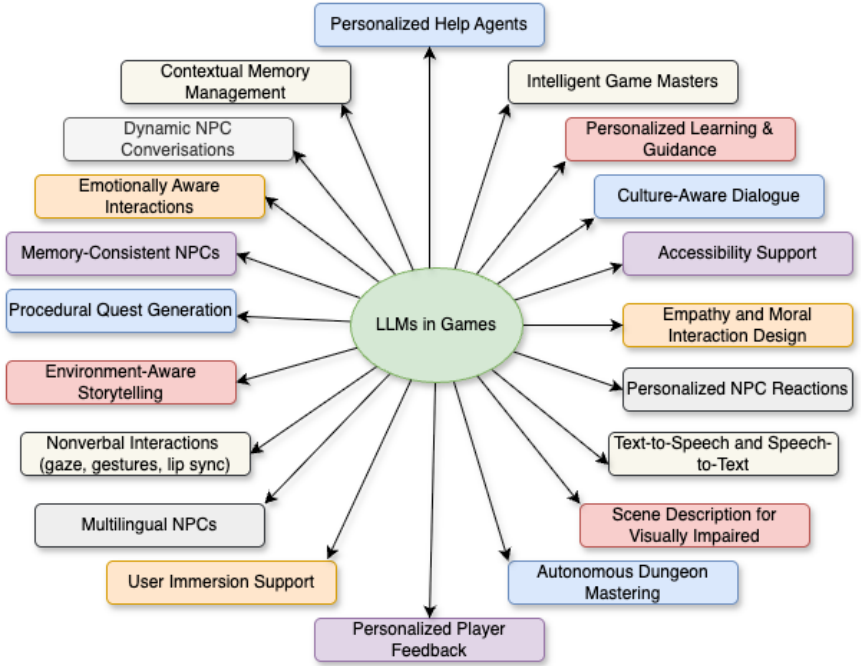
- **2024: Open-Source Acceleration**
  - Meta released LLaMA 3 (8B and 70B), improving open-source access to GPT-4 level models.
  - DeepSeek-V2 and Mixtral gained popularity for strong math and code reasoning abilities.

- **2025: Gemini 1.5 and Claude 3**
  - Google's Gemini 1.5 offered 1M token context, advancing long-context interaction and code capabilities.
  - Anthropic's Claude 3 improved logic, summarization, and safety even further.

Why integrate LLMs into games and virtual worlds?

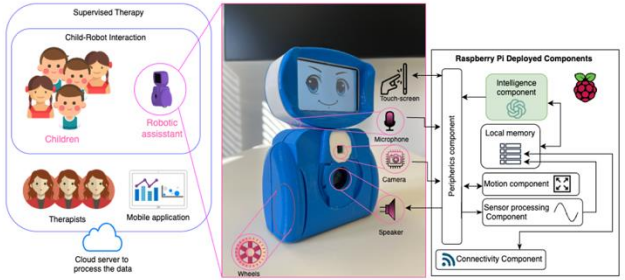
- ✓ **Intelligent Behaviors**
  AI characters can “reason”, adapt, and respond meaningfully to dynamic in-game events and player behavior.
- ✓ **Emotional & Personalized Experiences**
  Personalized dialogue and emotional understanding increase immersion.
- ✓ **Procedural Storytelling**
  LLMs support quest generation, branching narratives, and player-driven storytelling.
- ✓ **Puzzle Solving & Tutoring**
  In education or puzzle-based games, LLMs can act as in-game tutors, or adaptive helpers.
- ✓ **Accessibility & Localization**
  LLMs can dynamically translate or simplify content, making games more inclusive.
- ✓ **And many more…**

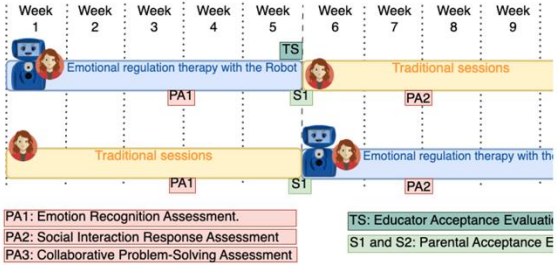
Why integrate LLMs into games and virtual worlds?

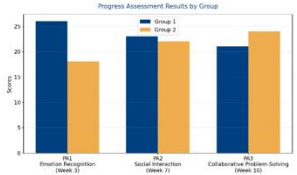


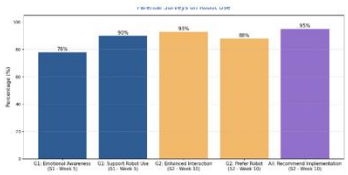
Why integrate LLMs into games and virtual worlds?

Supporting Preschool Emotional Development with AI-Powered Robots









- AI-powered robots are effective tools for enhancing emotional development in preschoolers.
- Future research will explore long-term impacts and refine interaction design based on child-robot trust dynamics and inclusion.

Paper

IDC 2025 - Reykjavík, Iceland

Santiago Berrezueta-Guzman, María Dolón-Poza, Stefan Wagner

ICVR 2025
International Conference on Virtual Reality

LLMs in Virtual Reality Games (based on research)

- A study was conducted in April 2025 at the Technical University of Munich exploring how LLMs can enhance player experience within VR environments.
- The research analyzes how LLMs enable intelligent, responsive, and emotionally aware interactions, transforming traditional NPCs into adaptive and context-aware virtual agents.
- This presentation highlights five key areas where LLMs can revolutionize VR games: Dynamic NPC Interactions, Procedural Storytelling, Intelligent Game Masters, Personalized Player Experience, and Accessibility.

How LLMs are Shaping the Future of Virtual Reality

Dynamic NPC Interactions

- LLMs enable emotionally intelligent NPCs that can adapt dialogue based on player tone, sentiment, and context.
- With LLM implementation like GPT-3 or 4, it is possible to generate facial expressions, gestures, and lip-sync aligned with speech.
- Emotionally expressive NPCs increase player immersion and produce nuanced psychological responses.
- Memory-aware systems like these ensure long-term consistency in conversations with the NPCs.
- Multimodal interaction through voice, gaze, and gesture further humanizes NPC behavior, improving believability and trust.

Procedural Storytelling

- LLMs can generate adaptive, branching narratives that evolve with player decisions in real time.
- Systems like Quest-GPT-2 and PANGeA demonstrate LLMs' ability to create diverse quests and dynamic scenes.
- Scene-aware storytelling allows characters to reference the environment and deliver spatially grounded dialogue.
- Persona-based dialogue systems help maintain character identity and emotional consistency.

Intelligent Game Masters

- LLMs can function as dynamic AI game masters, narrating, adjusting scenes, and improvising with players.
- They support the game immersiveness by managing rules, tracking progress, and adapting challenges.
- Players prefer emotionally supportive and cooperative AI Dungeon Masters, which enhances trust and engagement.
- Real-time assistants in VR show how LLMs can guide players without breaking immersion, both in entertainment and serious games.

Personalized Player Experience

- LLMs support personalization through adaptive dialogue, emotionally tailored narratives, and memory of prior interactions.
- Voice-based NPCs and human-like avatars improve engagement and make players feel “seen” by the game.
- Familiar-looking NPCs or avatars that reflect user preferences improve comfort and immersion.

Accessibility, Inclusivity, and Usability

- Spoken scene descriptions enable visually impaired players to navigate VR spaces.
- Educational tools with LLMs help autistic learners practice communication and learn independently.
- Inclusive design is supported through personalized difficulty, simplified dialogue, and cultural adaptation.
- Hand tracking and natural input methods also boost usability for less tech-savvy users.

Plugin Spotlight: Tools for LLM Integration in UE5

Through our research, we observed that there are multiple approaches to bringing LLM capabilities into Unreal Engine environments. While the language generation core can be handled by models like GPT-3.5 Turbo, enabling real-time interaction, emotional response, and immersive behaviors requires a robust plugin ecosystem.

We began by adding a chatbox using OpenAI's **GPT-3.5 Turbo and GPT-4** via the **VaREST** plugin and are currently working on implementing text-to-speech, lip sync, and multimodal feedback.

Here are some Unreal Engine 5.5 plugins we found particularly useful for integrating LLMs in VR development:

Plugin Spotlight: Tools for LLM Integration in UE5

HTTP GPT

A lightweight REST-based integration layer that communicates directly with GPT endpoints. Ideal for quick prototyping and testing GPT-based responses within Unreal Engine.

Runtime AI Chatbot Integrator

Enables embedding of real-time conversational agents in gameplay. Compatible with behavior trees and animation systems, allowing adaptive and immersive interactions.

Plugin Spotlight: Tools for LLM Integration in UE5

Runtime Speech Recognizer

Provides voice input capabilities without relying on cloud services. Useful for creating responsive, low-latency Unreal Engine projects with voice commands and local speech understanding.

VaREST

A widely adopted plugin for handling HTTP and REST API requests in Unreal. Popular for general-purpose API integration due to its flexibility and ease of use.

From Text to Presence: What Makes an NPC Feel Real?

Text-based interaction is only the foundation—intelligence goes far beyond.

While basic LLM integration allows text-based conversation with NPCs, **a truly intelligent virtual character** needs much more:

- ❑ **Text-to-speech (TTS)** & **Speech-to-text (STT)**
- ❑ **Lip-syncing** based on what's being said
- ❑ **Personality & backstory memory**
- ❑ **Facial gestures, emotional expression**
- ❑ **Interaction with nearby objects & context awareness**
- ❑ **Voice customization & emotional tones**

Convai - Conversational AI for Virtual Worlds

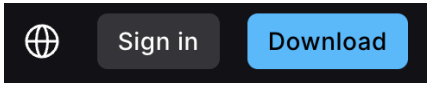
To bring these to life, today we'll use **Convai** — a powerful platform for building AI NPCs in games.

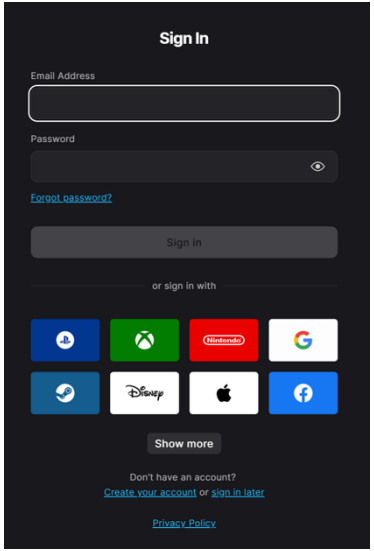
- Convai offers **free usage** for up to **50 monthly interactions.**
- Now let's integrate the plugin and create our **first Metahuman with real conversational abilities.**

Unreal Engine Setup - Installing UE5.5

1. Download Epic Games Launcher

- Go to epicgames.com
- Download and install the launcher
- Sign in with your account or create a new account





Unreal Engine Setup - Installing UE5.5

2. Install Unreal Engine 5.5

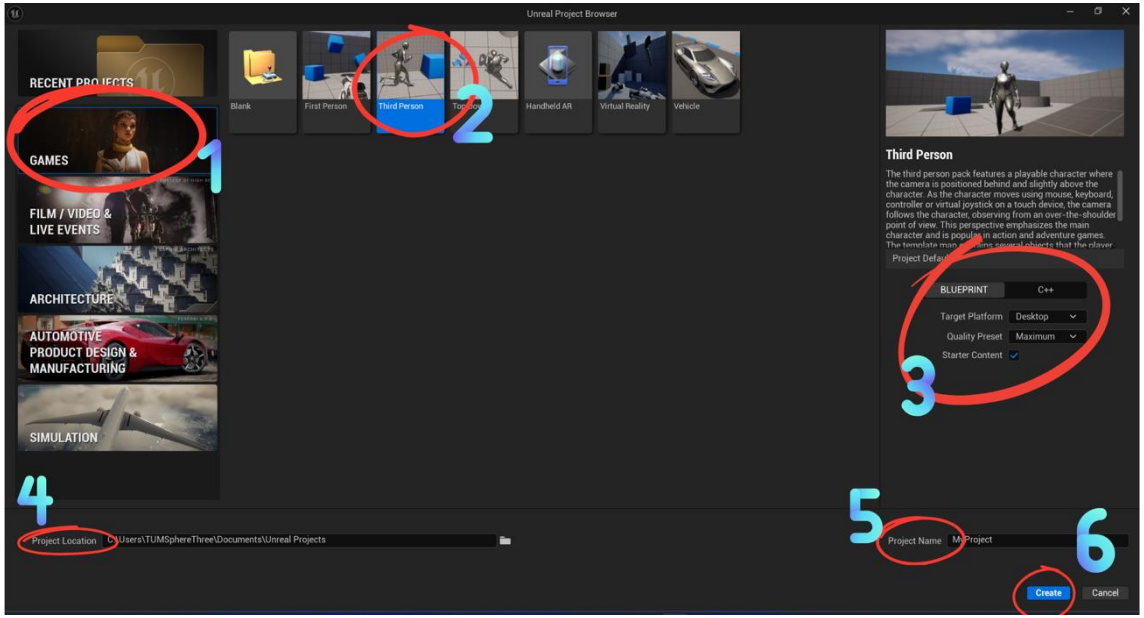
- Open the launcher
- Navigate to the **Unreal Engine**
- Select the **most recent version** (currently 5.5.4) and start the installation
- Choose the install location

Unreal Engine Setup - Project Initialization

3. Create a New Project

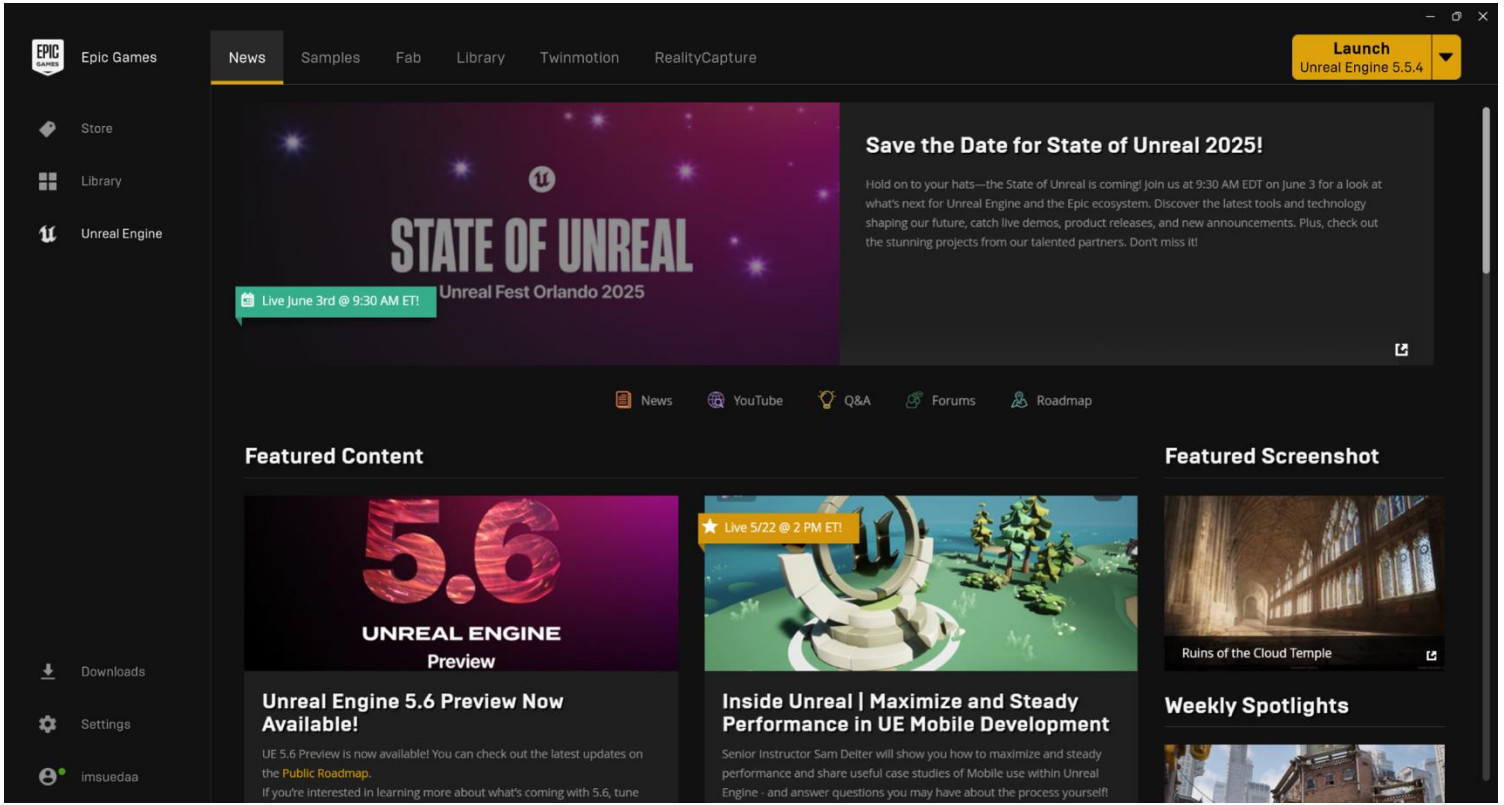
1) Select the **Games** category
2) Choose the **Third Person** template
3) Set Blueprint + Include **Starter Content** for testing
4) Choose your project folder
5) Name your project
6) Click **Create**

Steps to Create a New Project in Unreal Engine

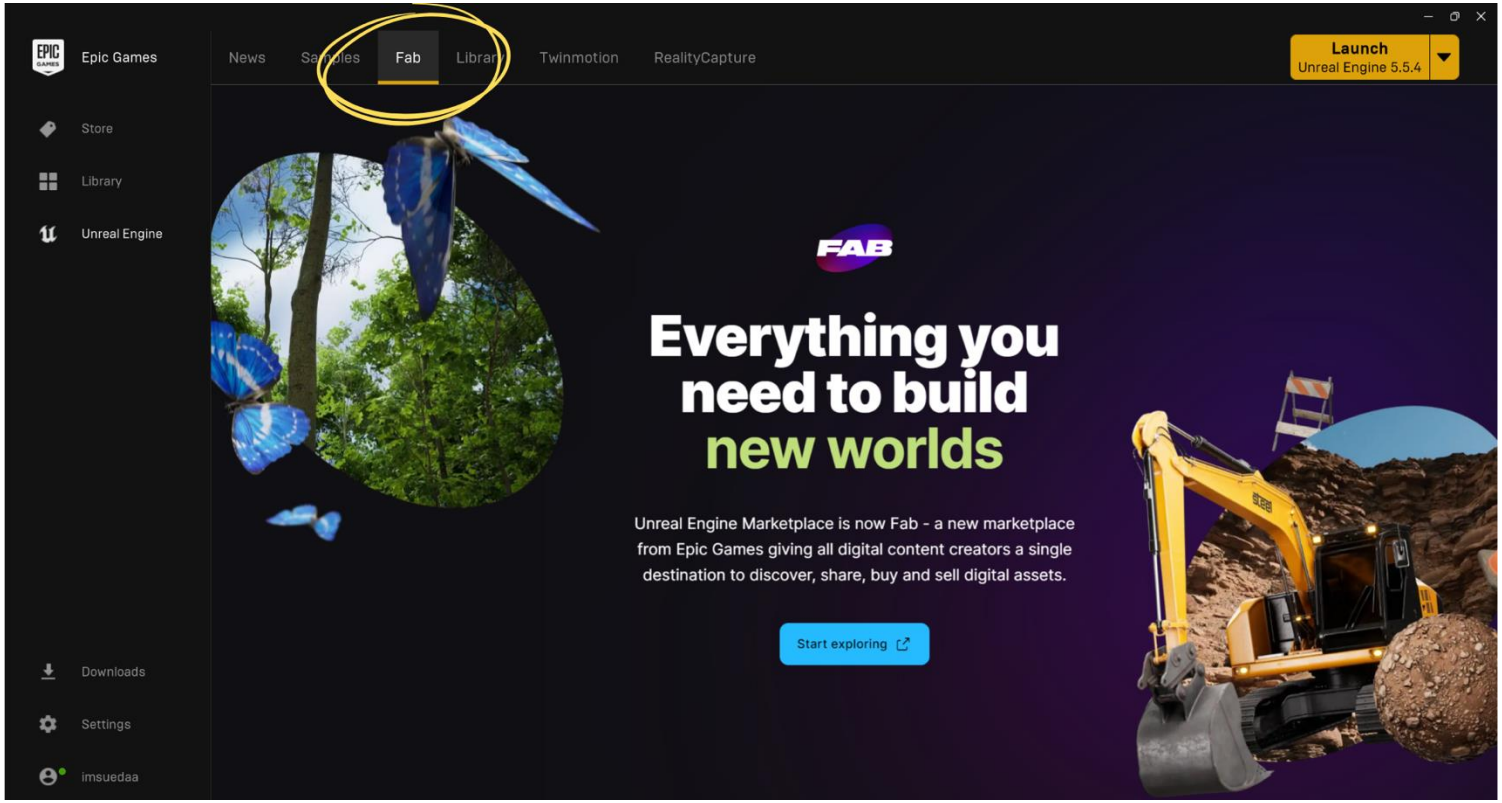


Now Let’s Download the Convai Plugin!

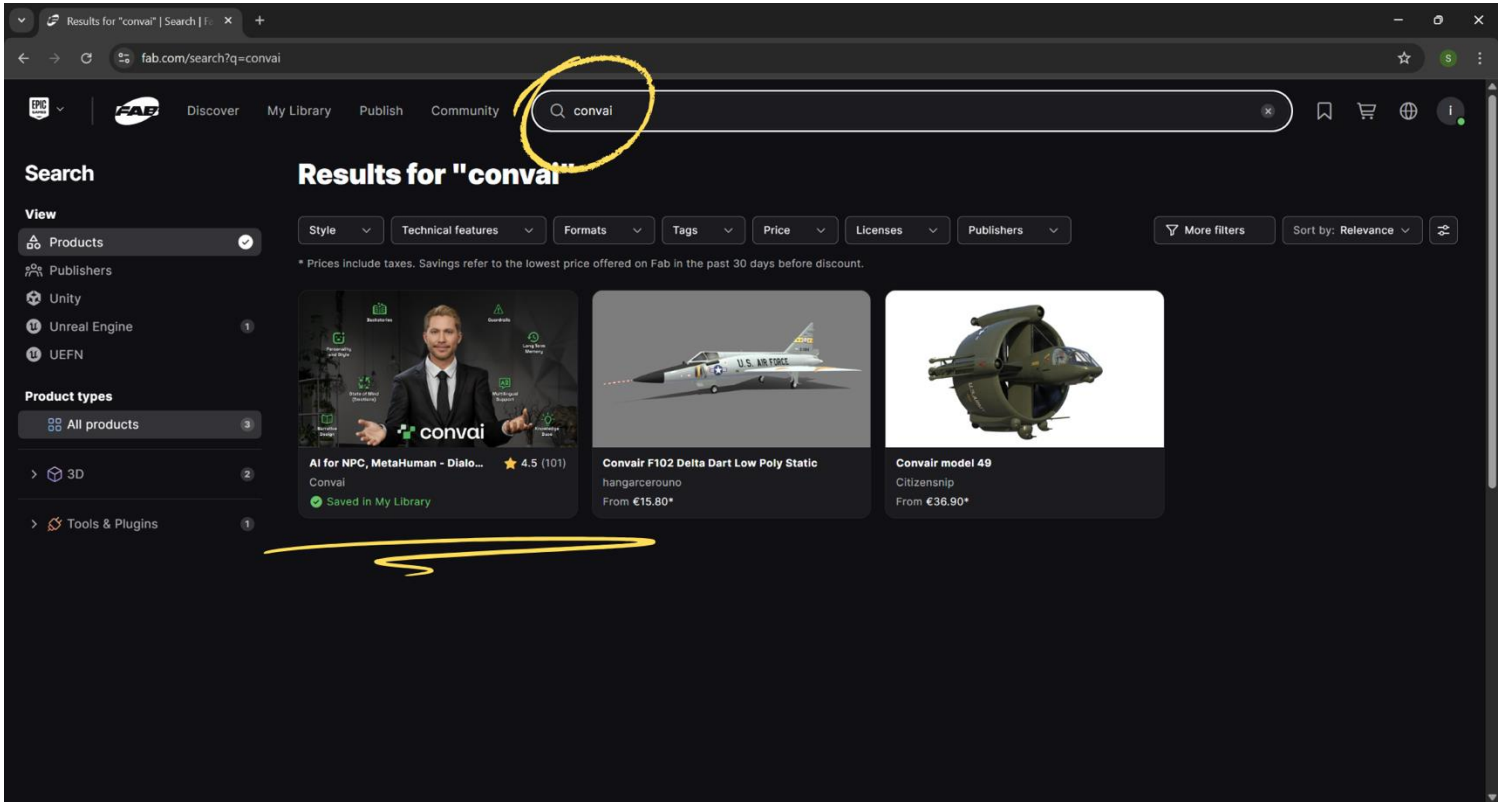
1. Go to the **Epic Games** app.



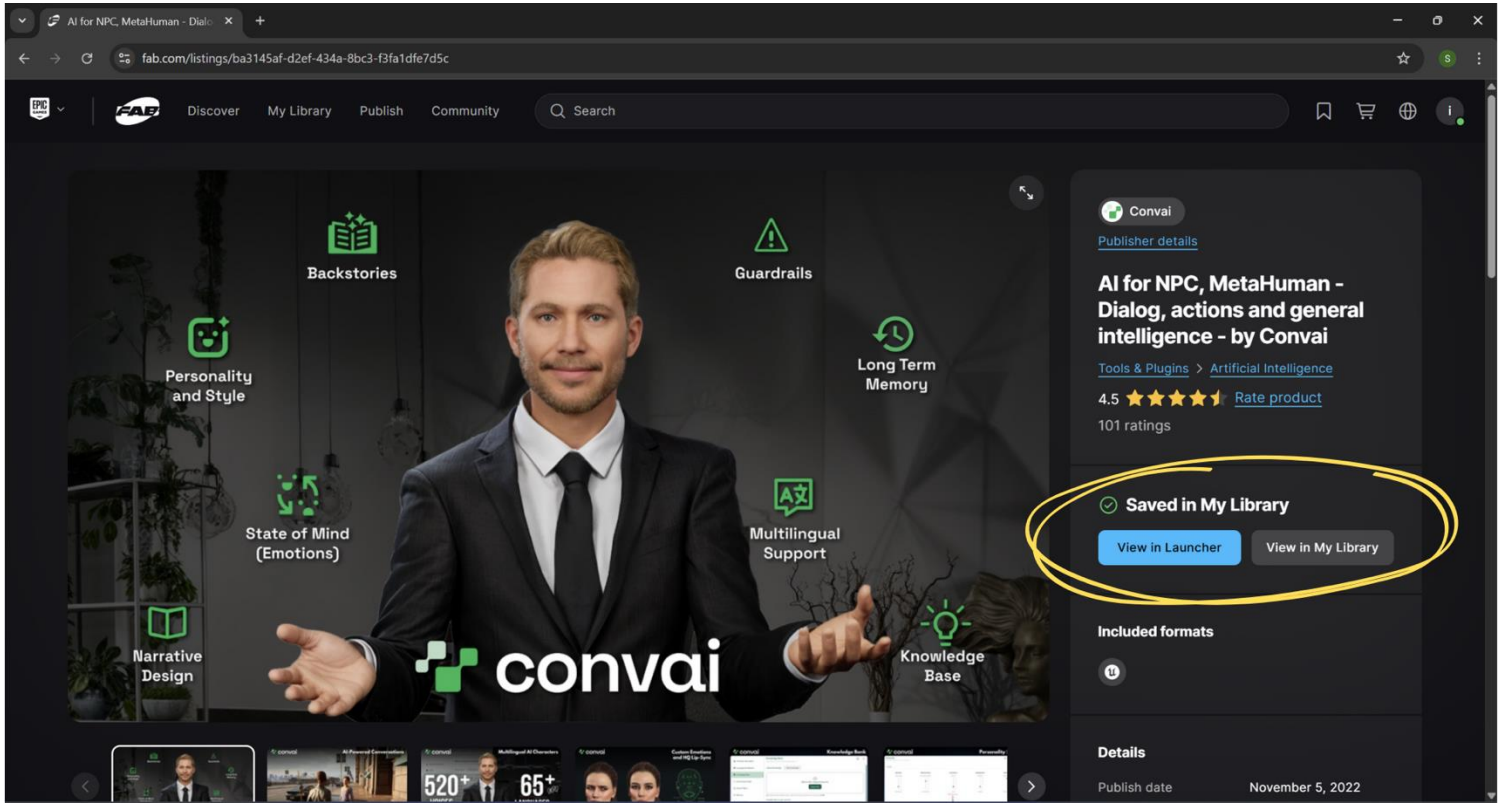
2. Click on the “**FAB**” tab from the top menu bar.



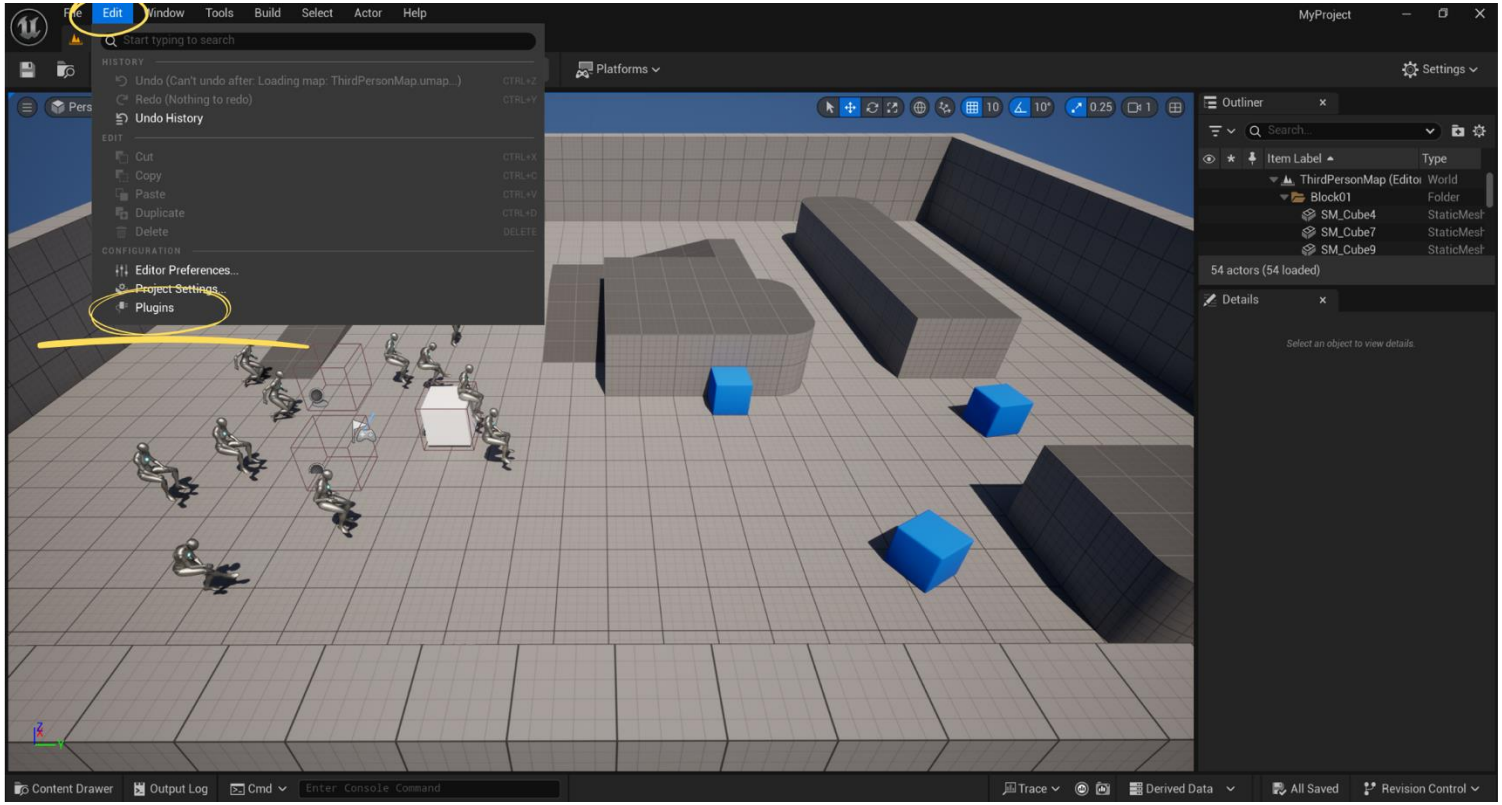
3. Type “**Convai**” into the search bar.



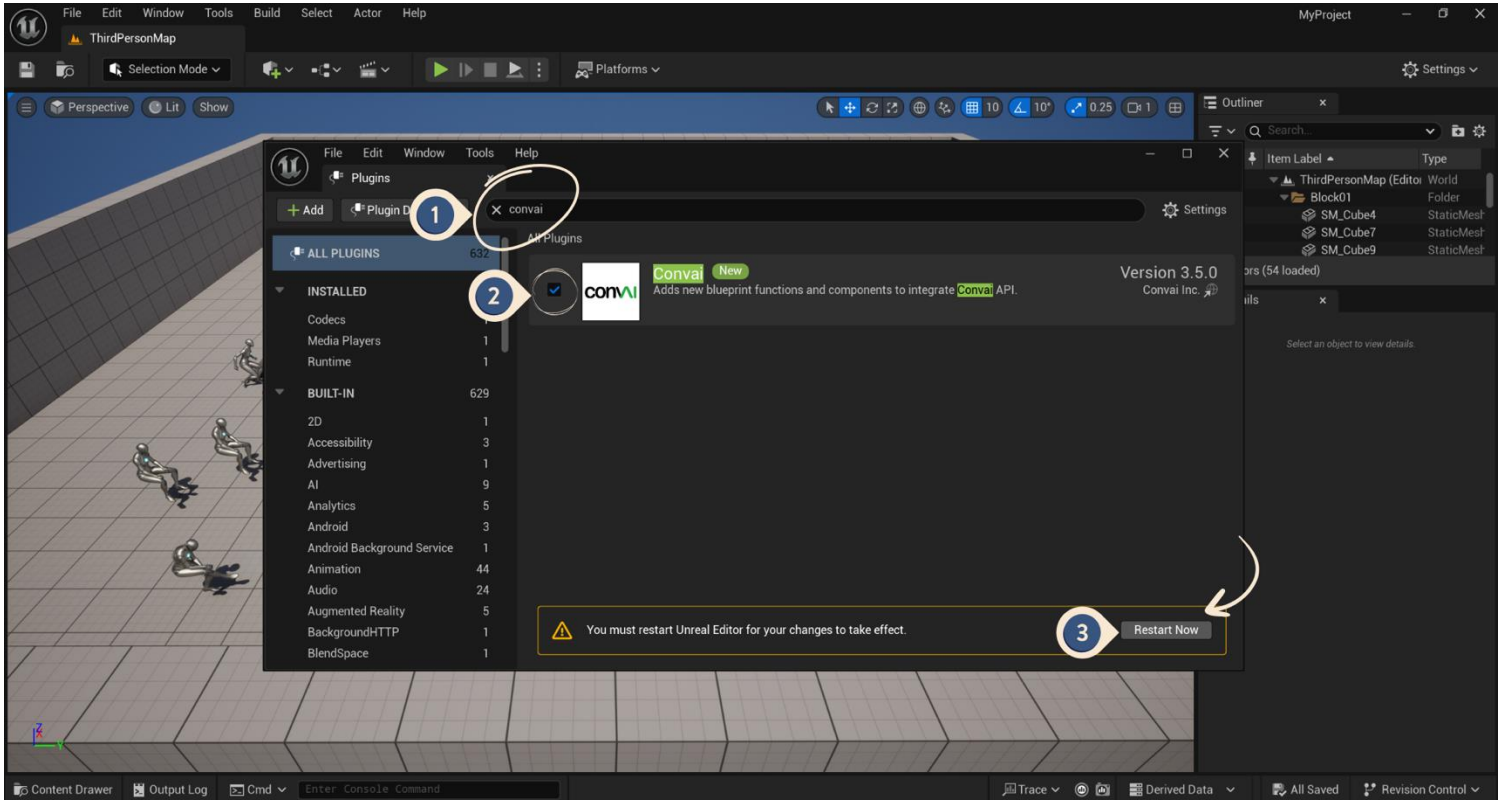
4. Click on it and **add it to the library.**



5. Open your Unreal Engine project. From the top menu, go to Edit → **Plugins**

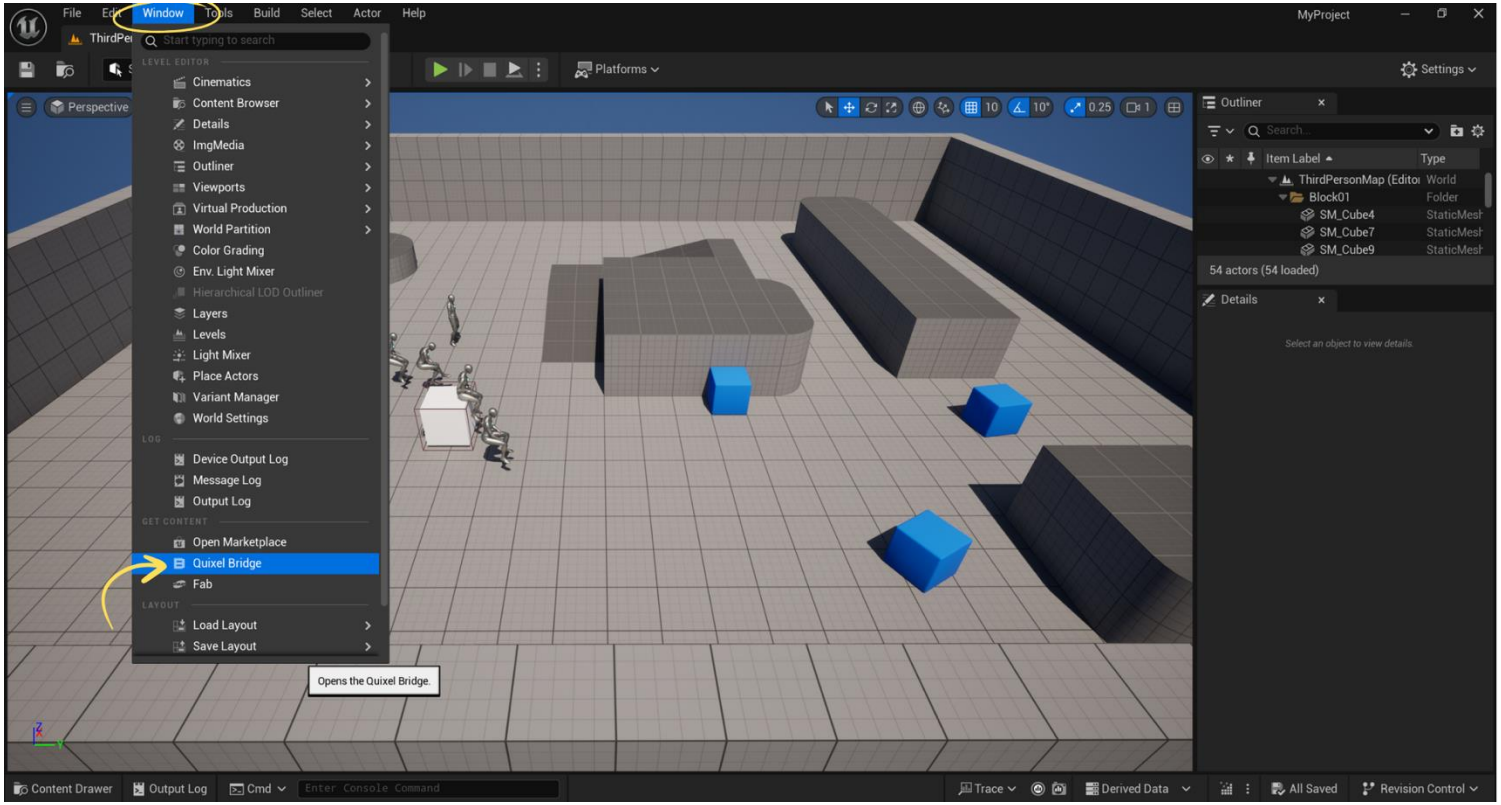


6. In the Plugins, search for **“Convai”**. Check the box, **restart** the editor.

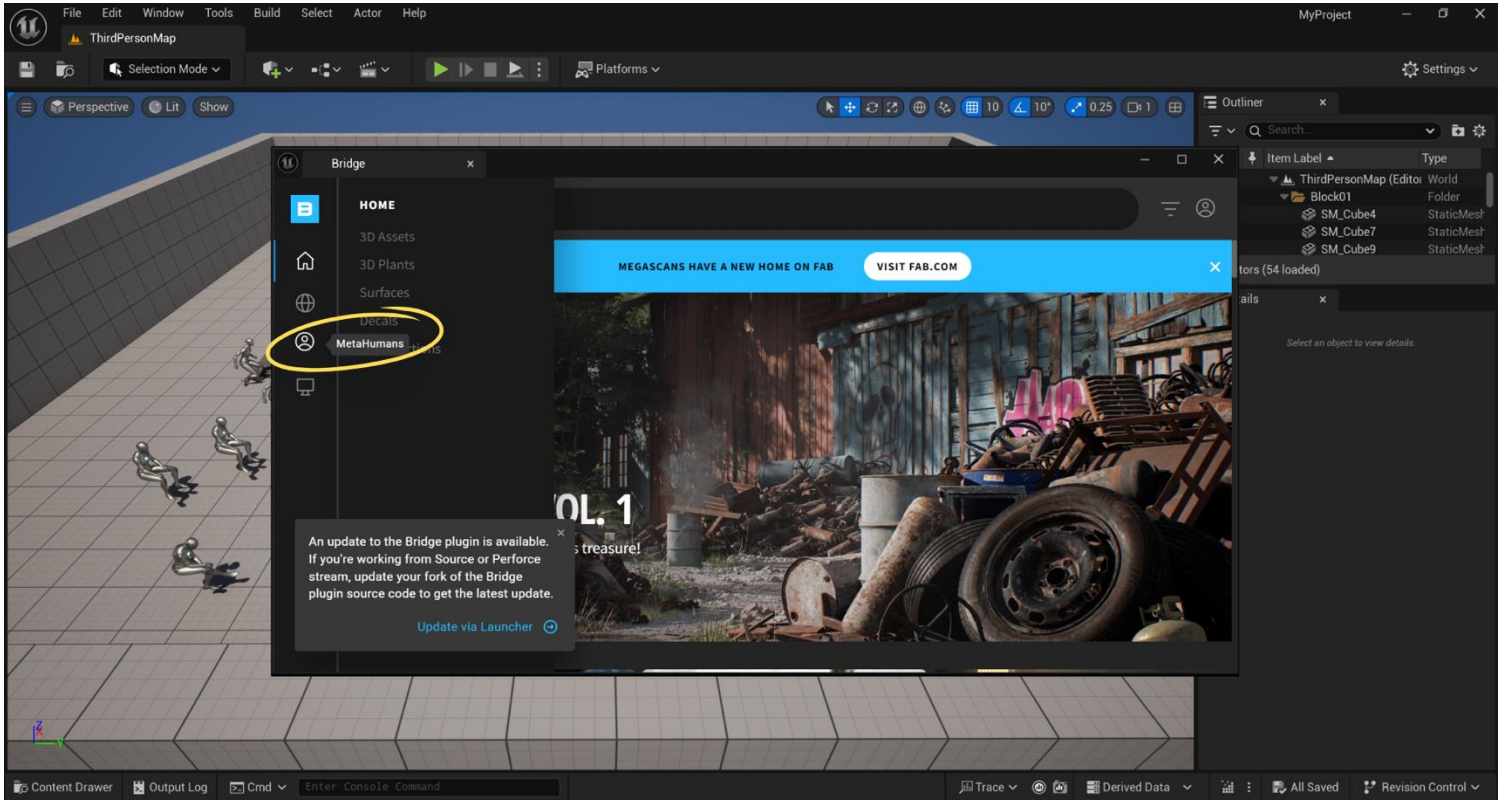


We enabled the plugin. Now it is time to add our first Metahuman to the scene!!

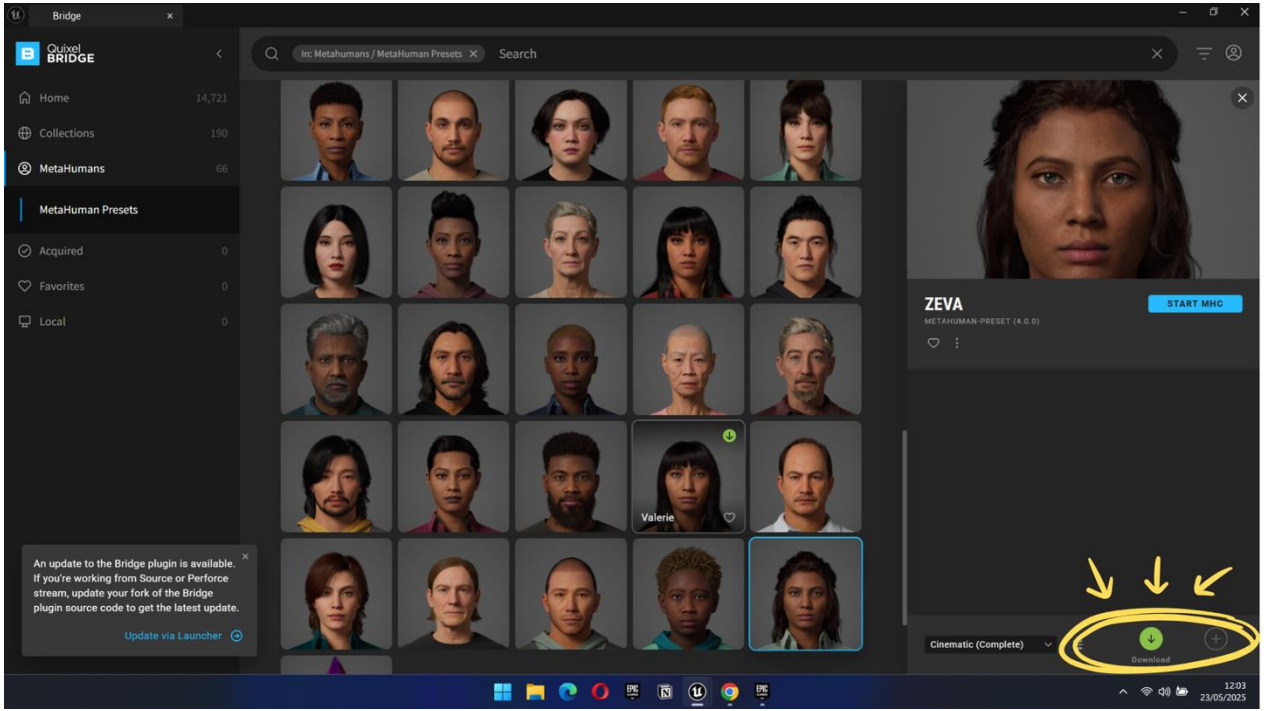
1. Open **Quixel Bridge** → Top menu: Window → Quixel Bridge



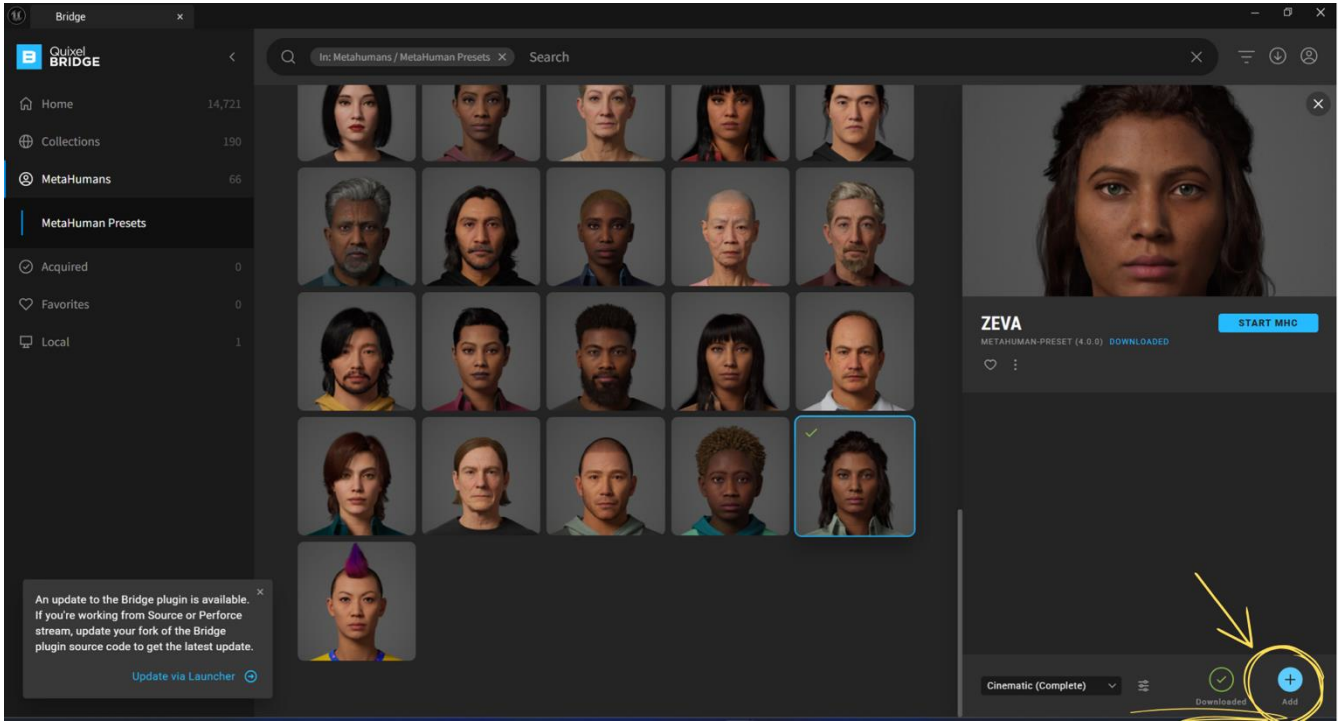
2. Go to **MetaHumans** tab.



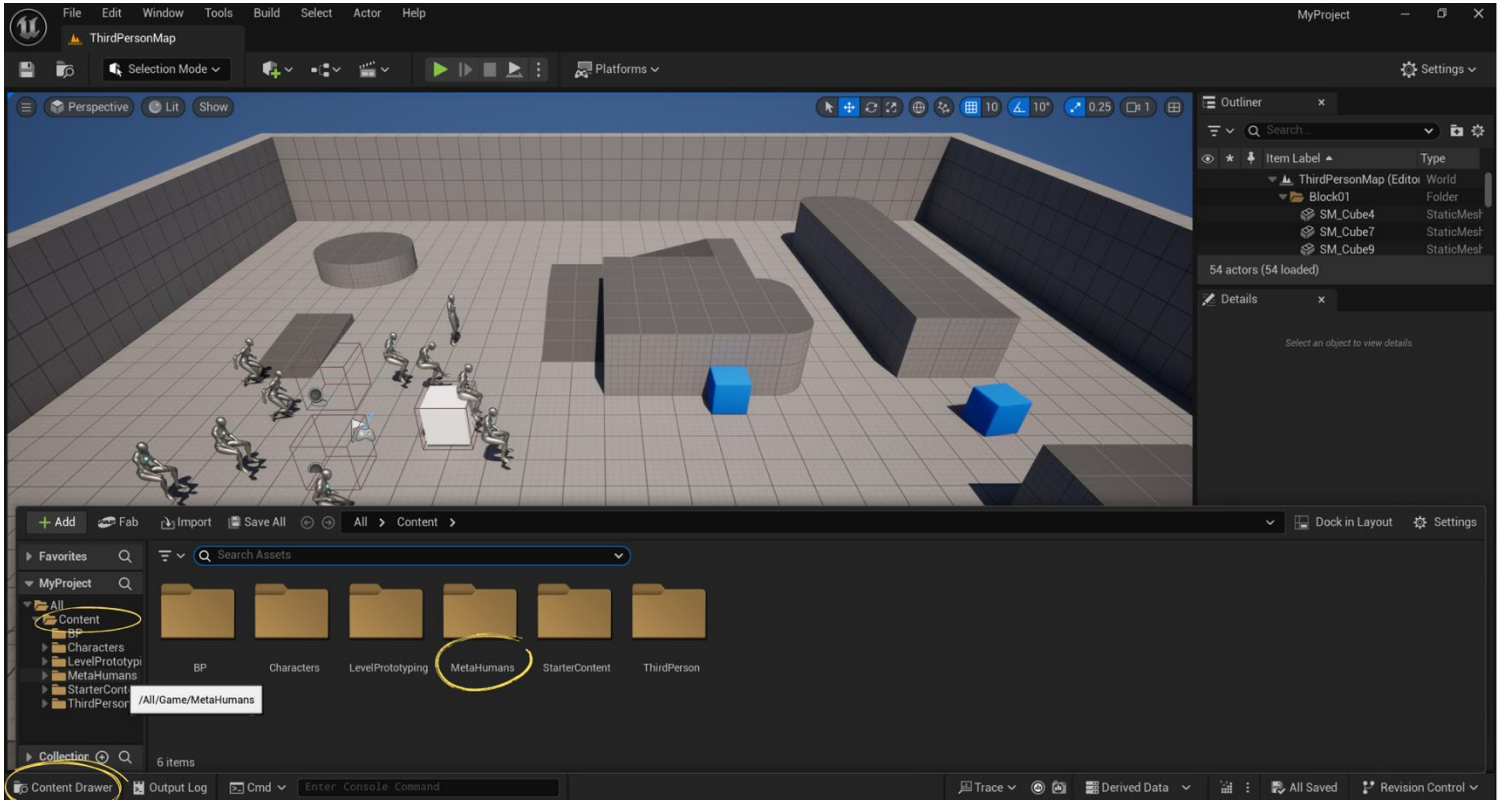
3. Choose a character → Click on a preset (e.g., “Zeva”), then click “**Download**”.



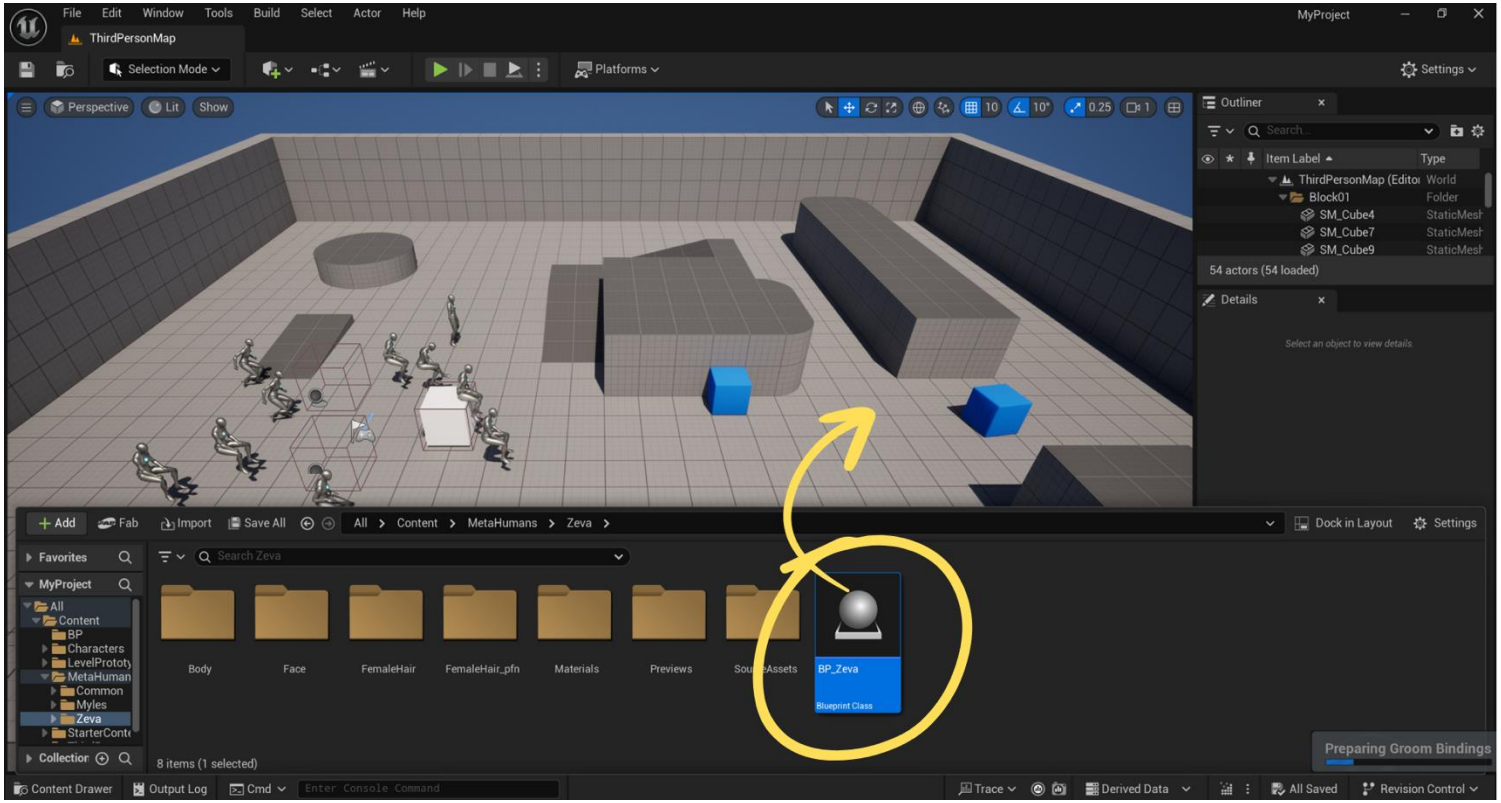
4. Once downloaded, click the “**Add**” button (This imports all assets into your project under /Content/MetaHumans/).



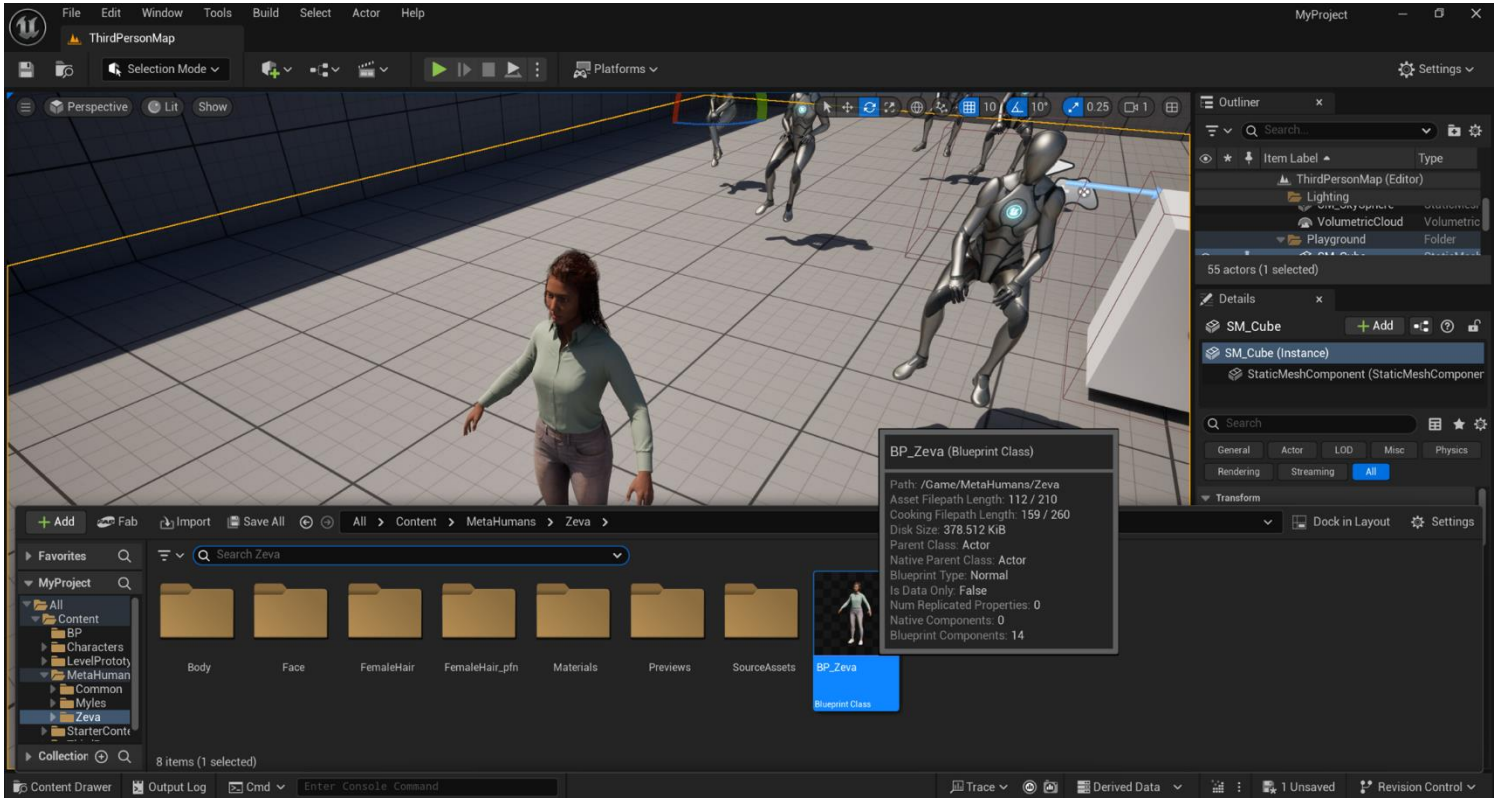
5. Go to **Content Drawer** → Content → MetaHumans.



6. **Drag and drop** “BP_YourCharacter” from the Content Drawer into the level.

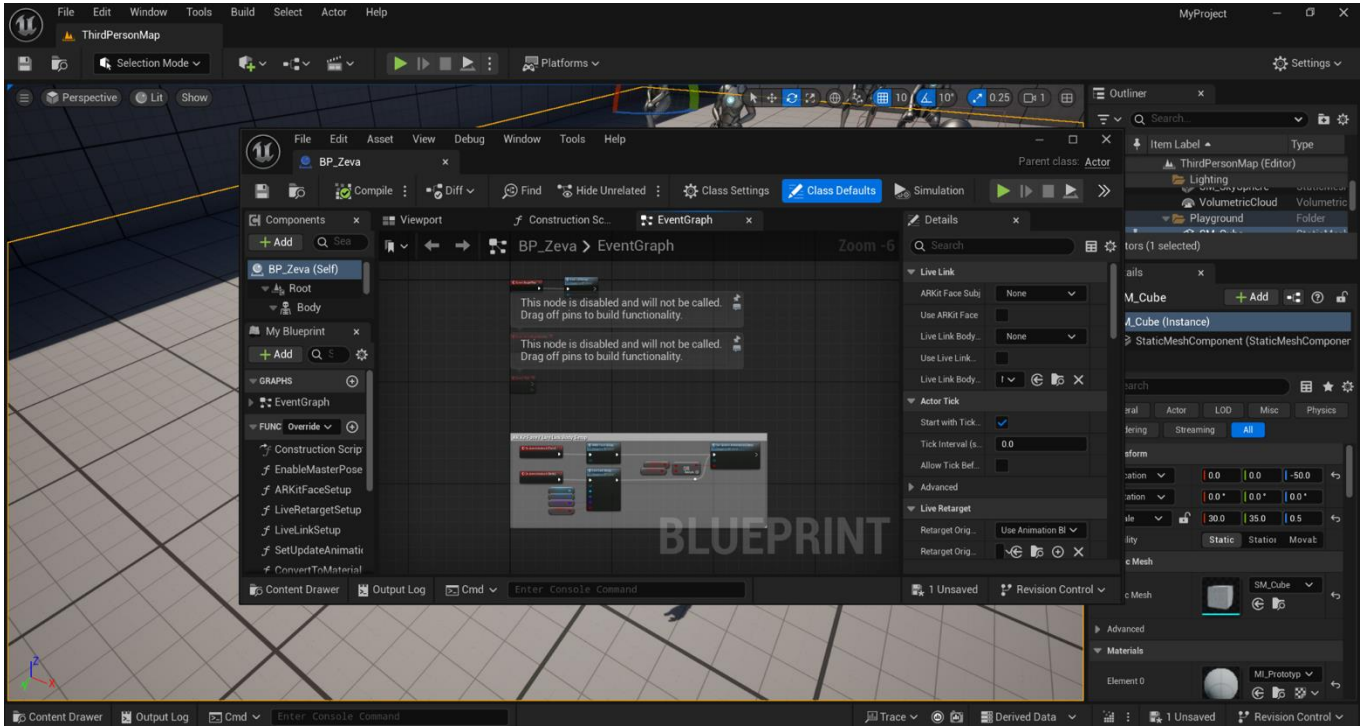


7. We now have a Metahuman in our scene, ready for Convai integration.

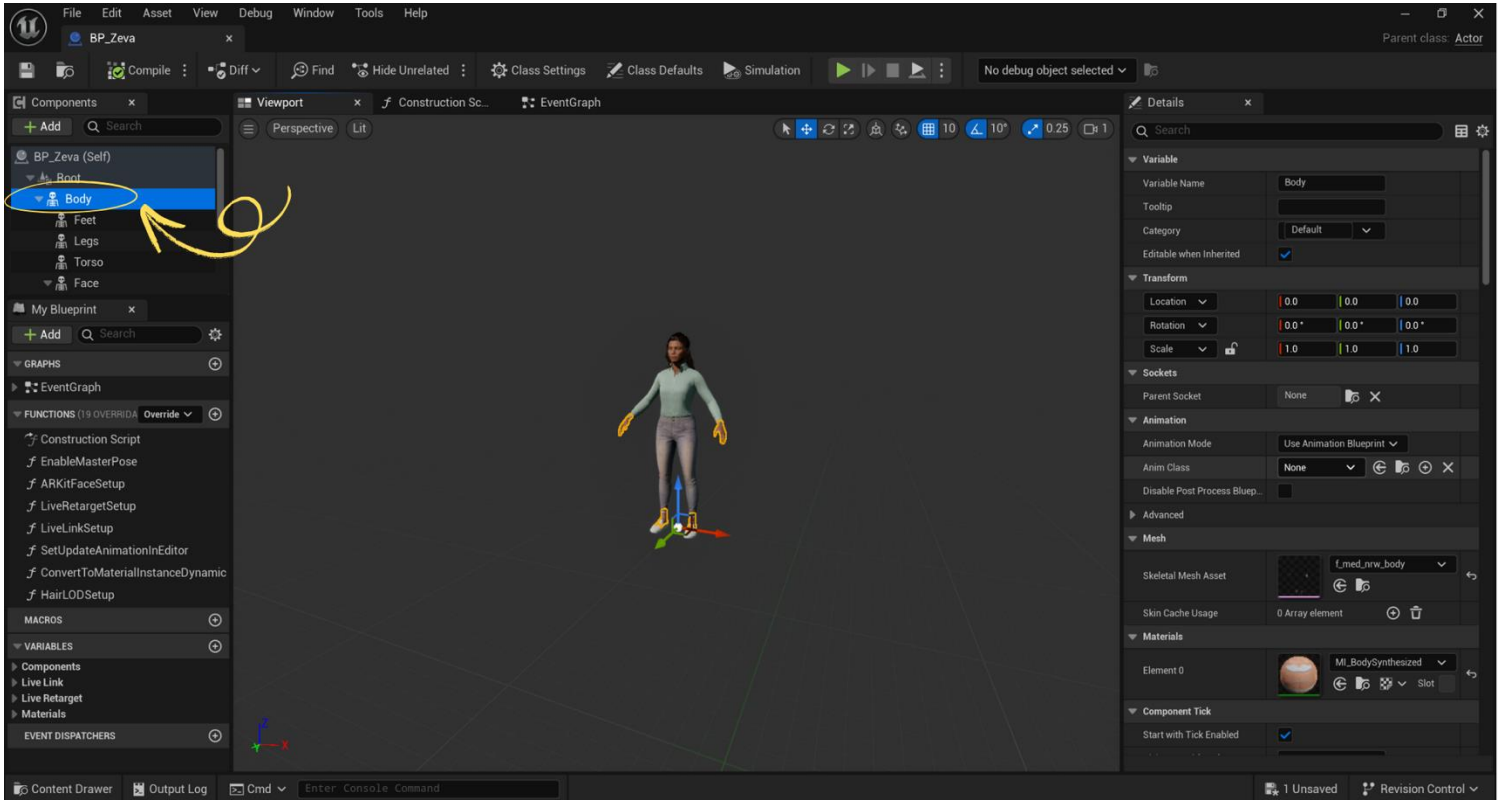


Now let’s add some animations to our character!

1. Navigate to your Metahuman asset and **double-click to open** it in the **Blueprint Editor**.

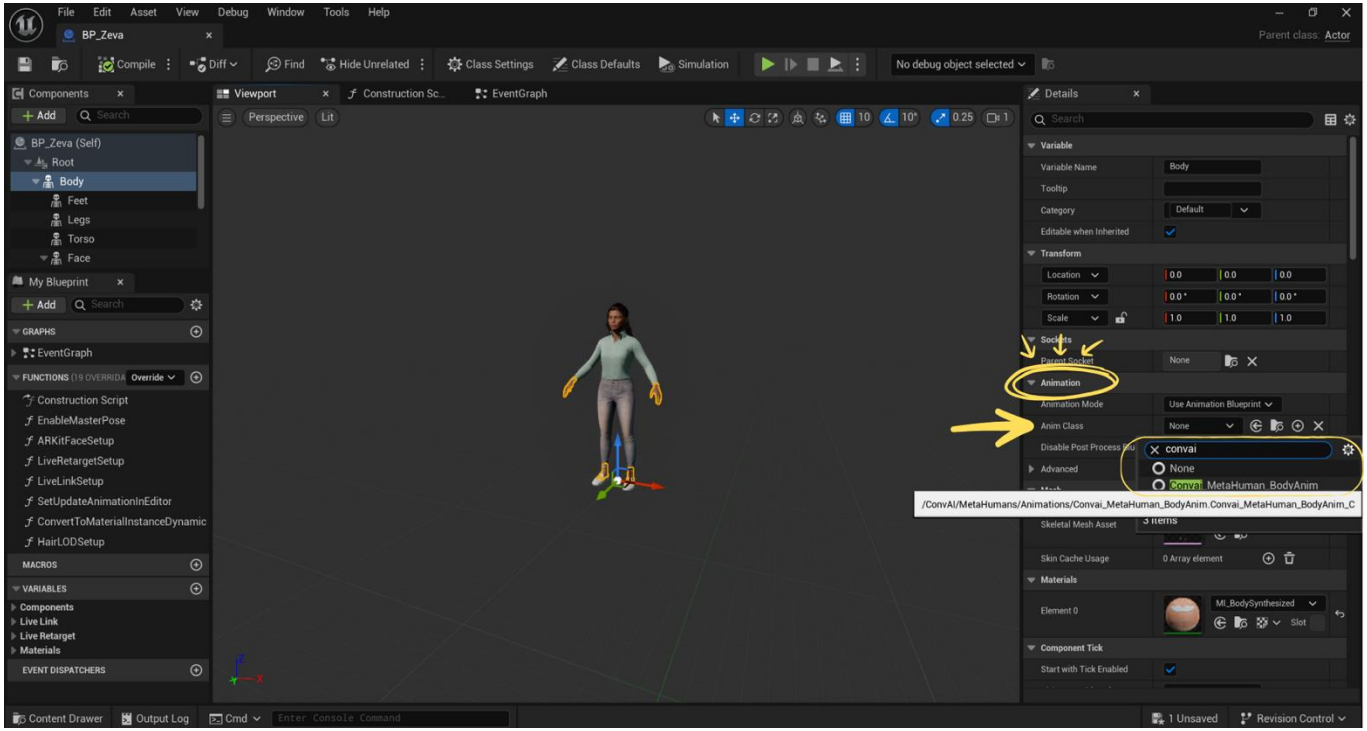


2. Click the **Viewport** tab & on **Body** component on the left.

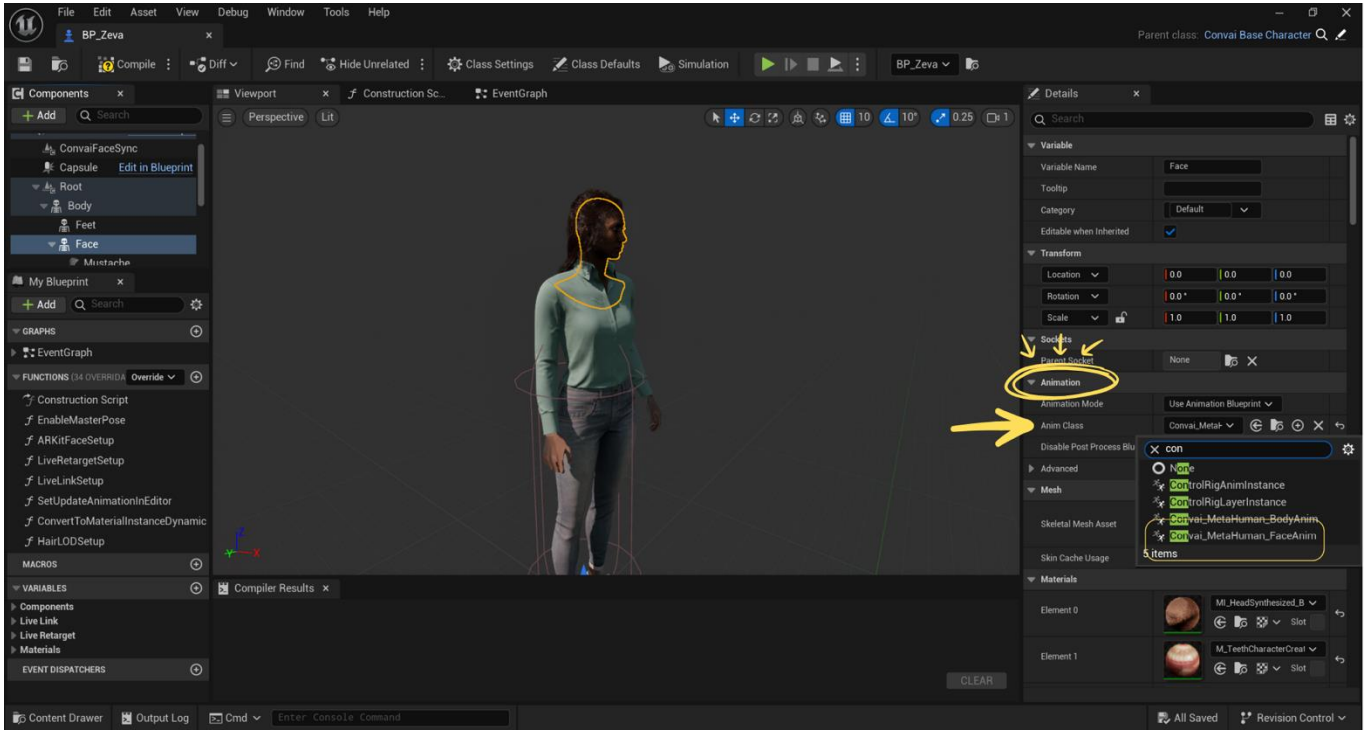


3. On the right panel, under **Animation** → **Anim Class**, choose:

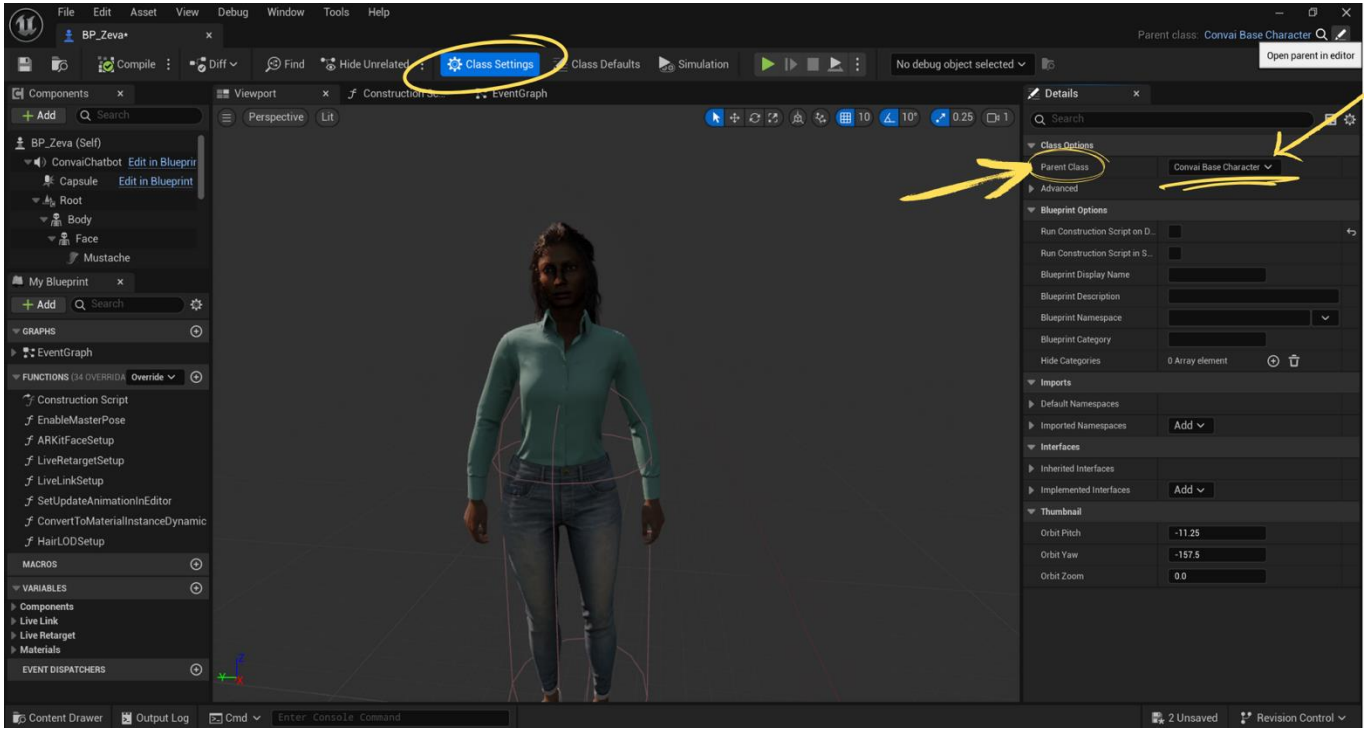
`Convai_MetaHuman_BodyAnim`.



4. Then we do the same for **Face** component. On the right panel, under **Animation** → **Anim Class**, choose: `Convai_MetaHuman_FaceAnim`.

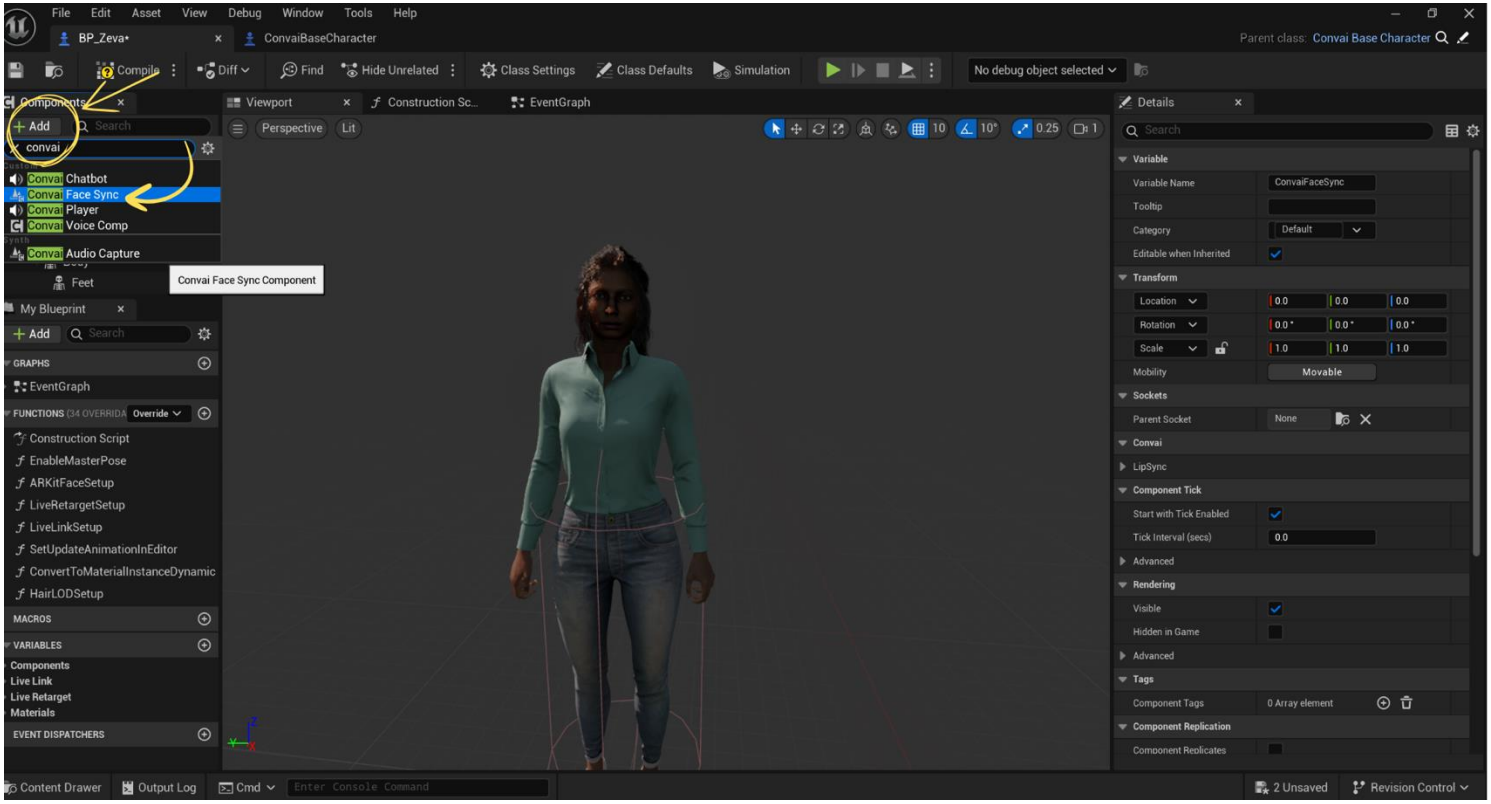


5. In the top bar, click **Class Settings**. On the right, find **“Parent Class”** and change it from `Actor` to: `ConvaiBaseCharacter`.



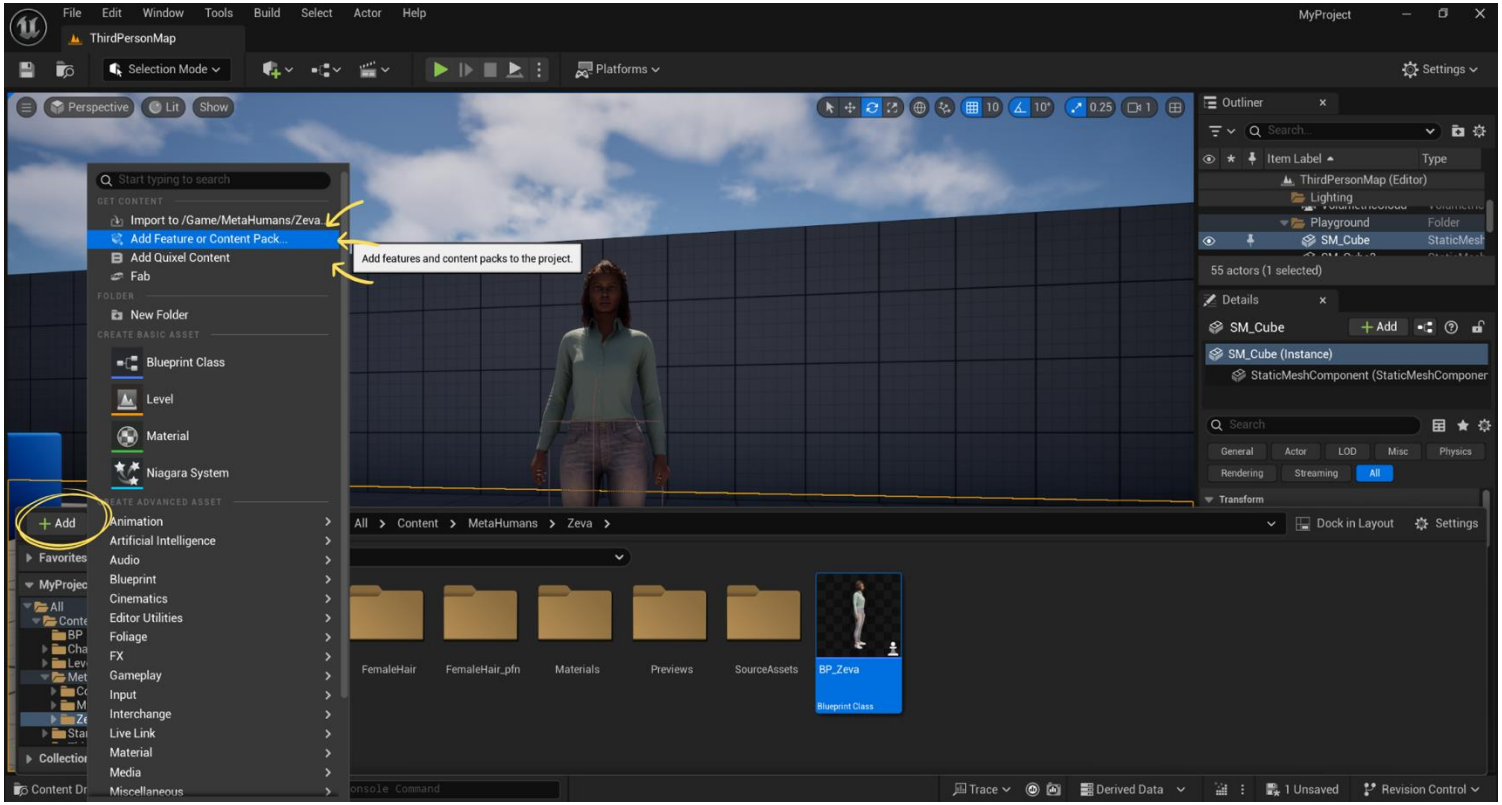
ConvaiBaseCharacter is a simple blueprint that includes several helpful functions to simplify and speed up your setup process.
If you're curious, you can explore its logic by clicking the **edit icon** in the top-right corner. Feel free to review, modify, or disable any built-in functions depending on your needs.
In this tutorial, we'll keep things simple and won't dive into its internals—our focus is on helping you get an **intelligent, talking Metahuman** up and running as quickly as possible.
Now let's add the **lip-sync**!

6. Click **Add** on the left, and then “**Convai Face Sync**”.

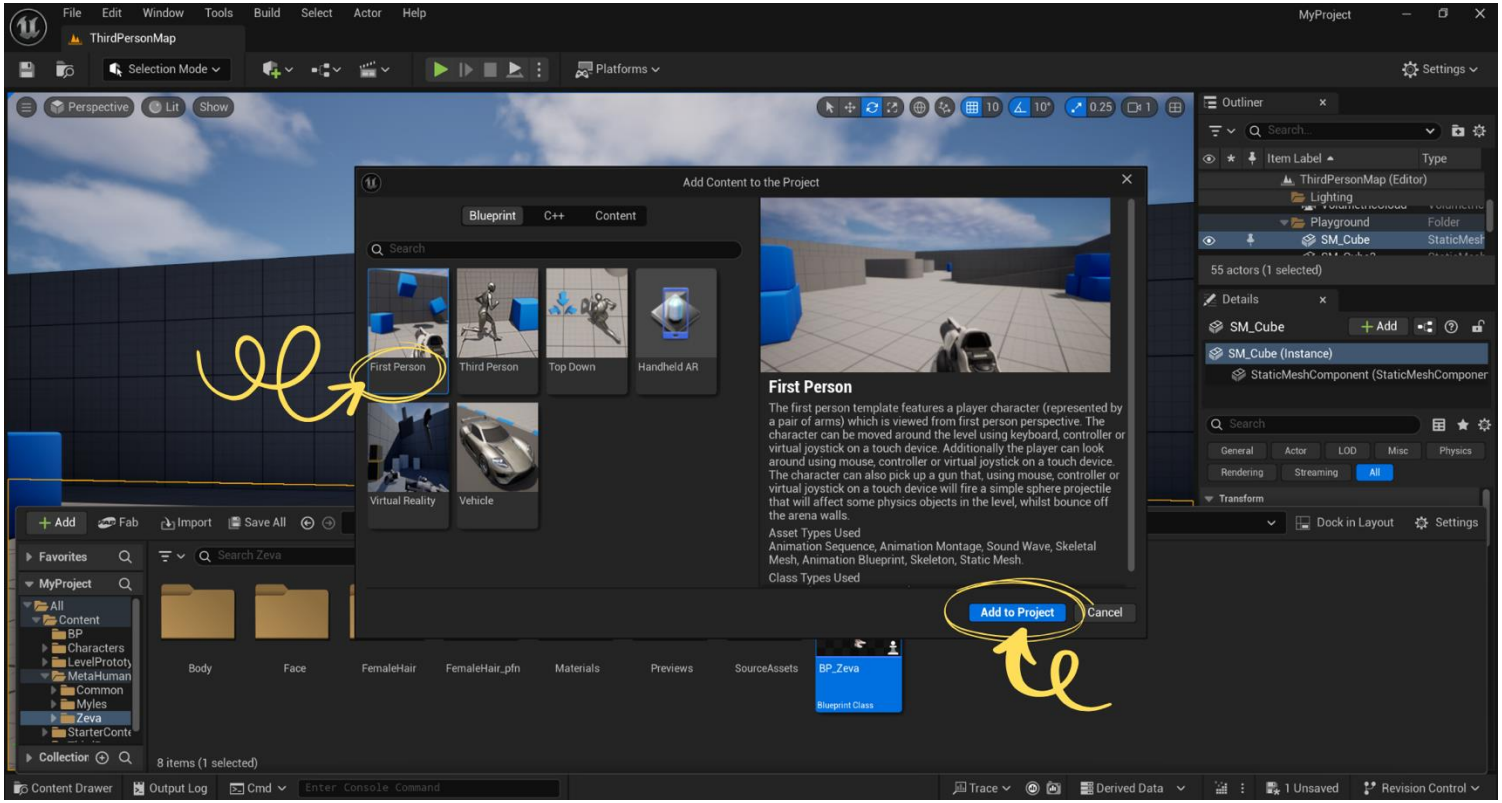


🎮 Now, Let’s Set Up Our Player Blueprint 🎮

1. Go to **Content Drawer** → **+Add** → Add Feature or Content Pack…

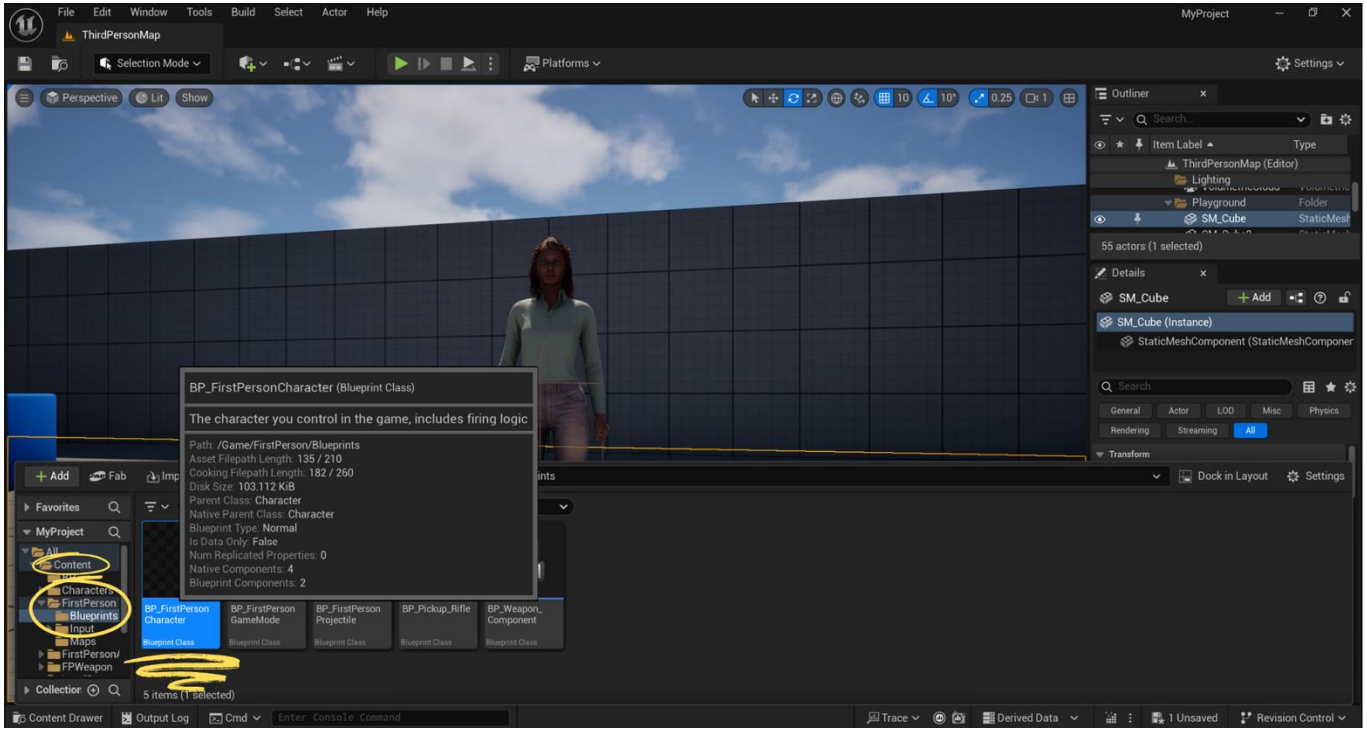


2. Let’s select **First Person** and click **Add to Project**.

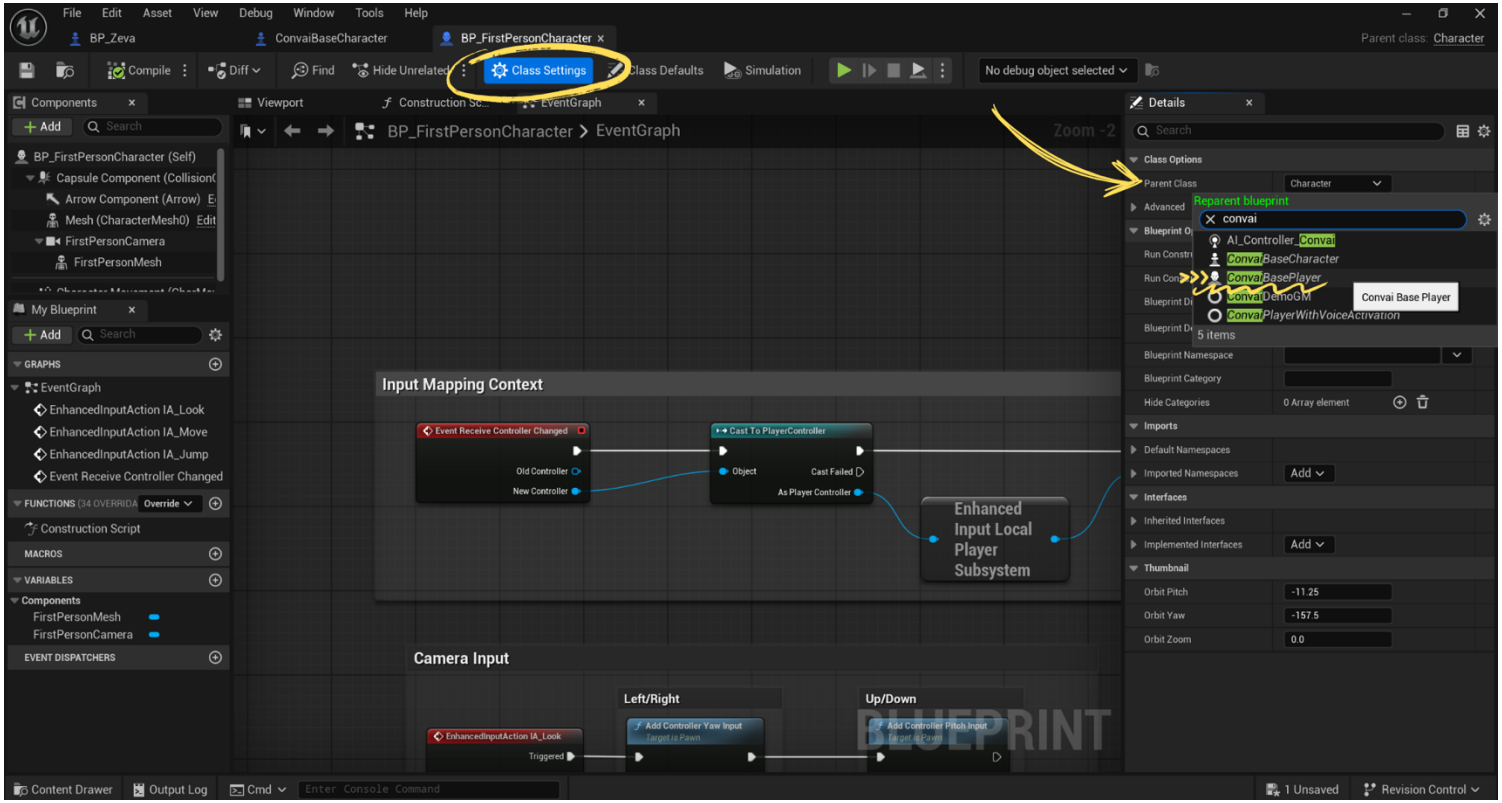


3. Go to **Content → FirstPerson → Blueprints**.

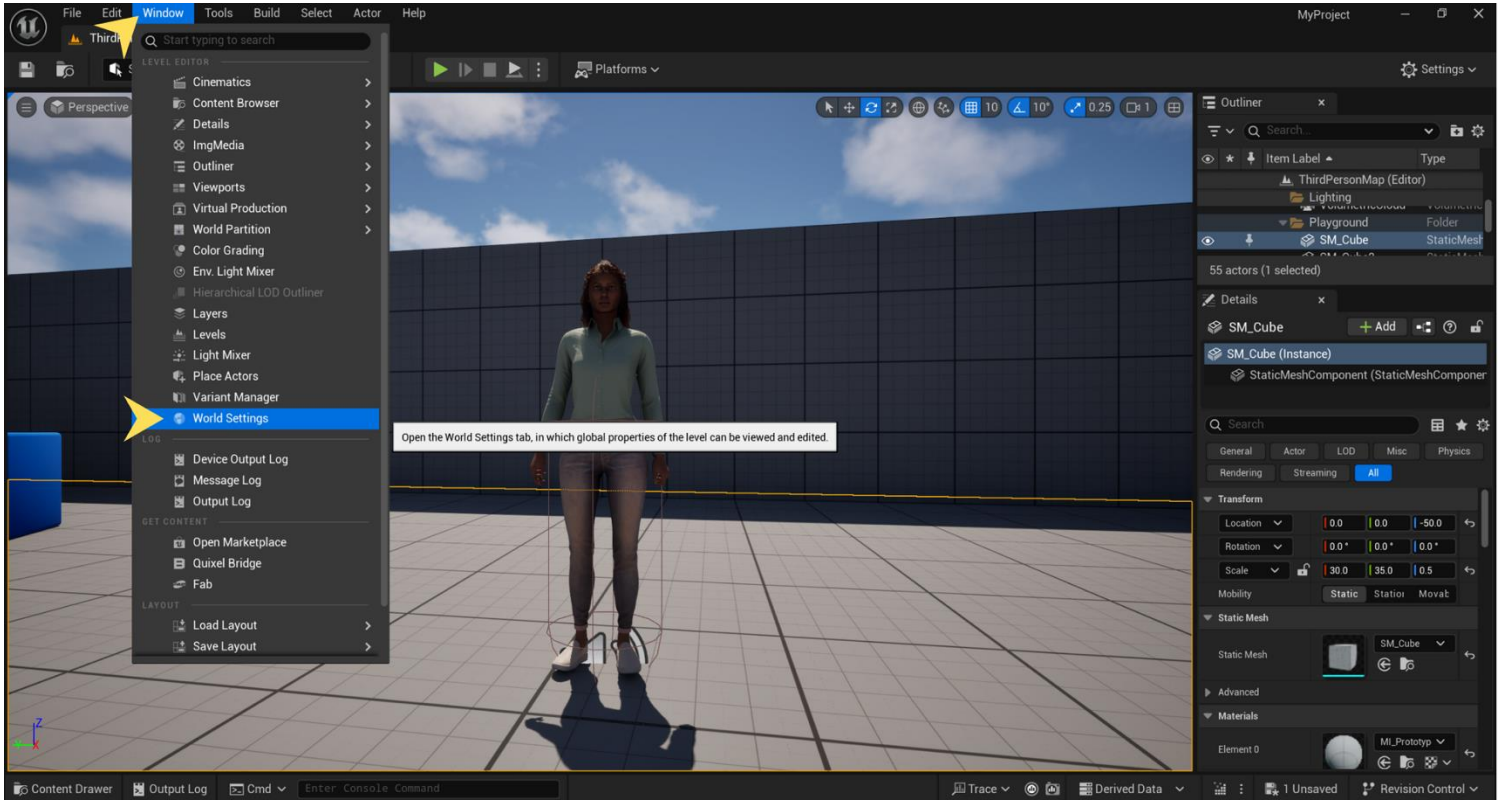
Double click on `BP_FirstPersonCharacter`



4.**Class Settings → Parent Class**, search for and select: `ConvaiBasePlayer`

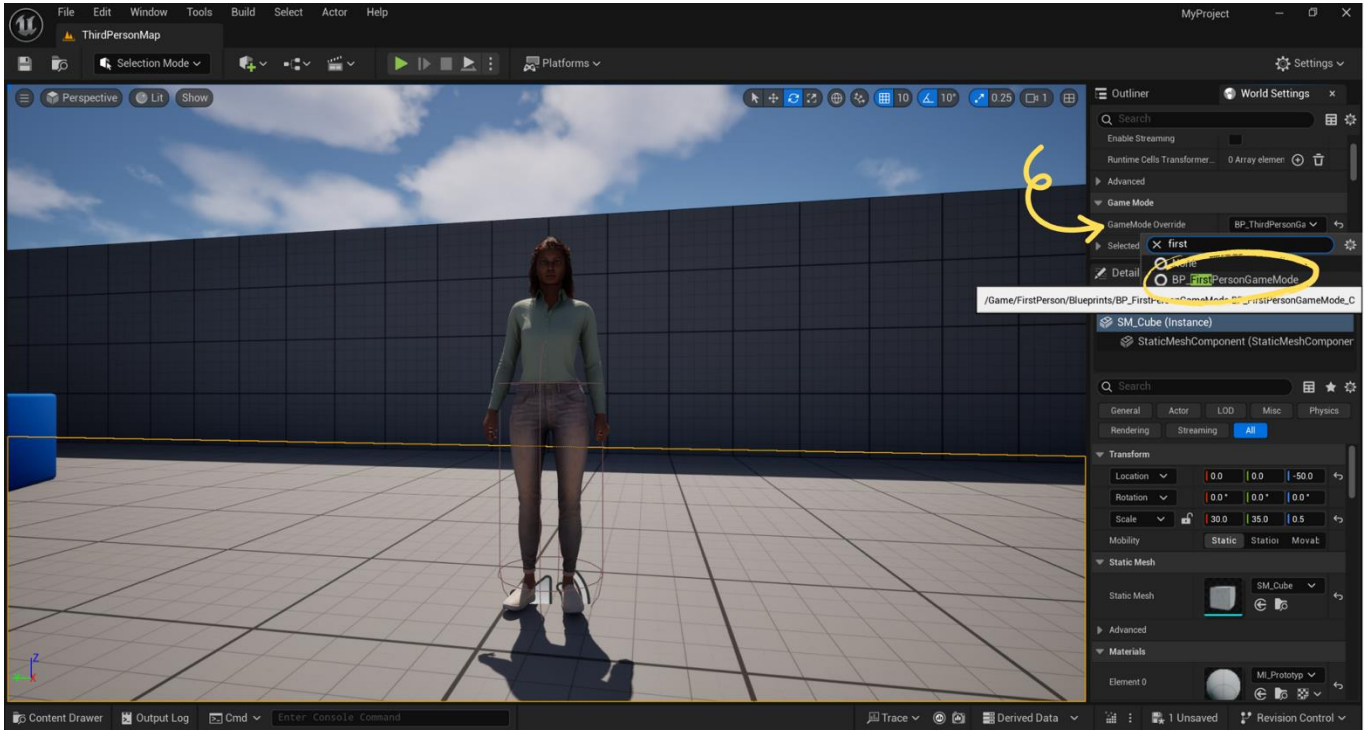


5. Navigate to **Window → World Settings**.



6. Go to **Game Mode** → set **Game Mode Override** to

`BP_FirstPersonCharacter`.



7. Now when you hit **Play**, you control a character that can speak, listen, and interact with LLM-powered NPCs in real time. Let's try it out!

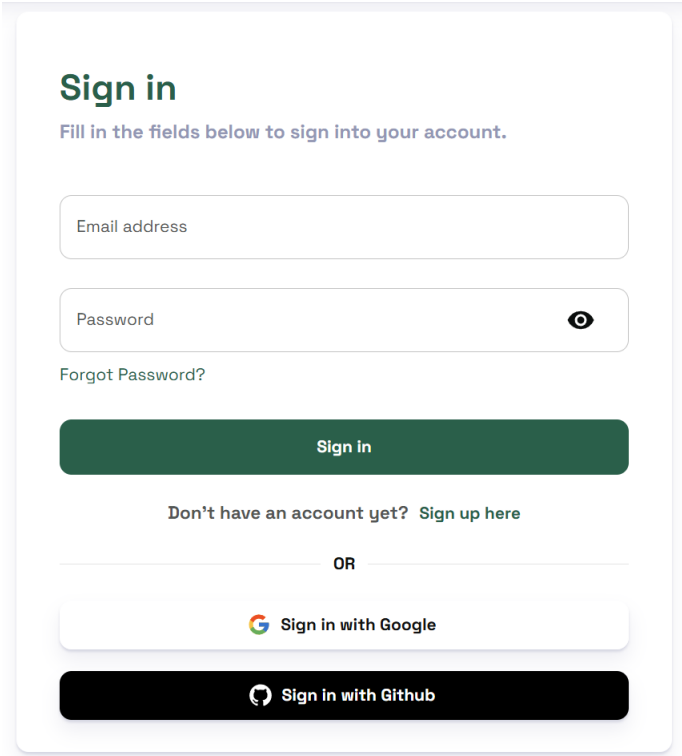
Getting Started with Convai

Now we'll head over to **convai.com** to define our character's **name**, **backstory**, **personality**, and even their **voice and accent**.

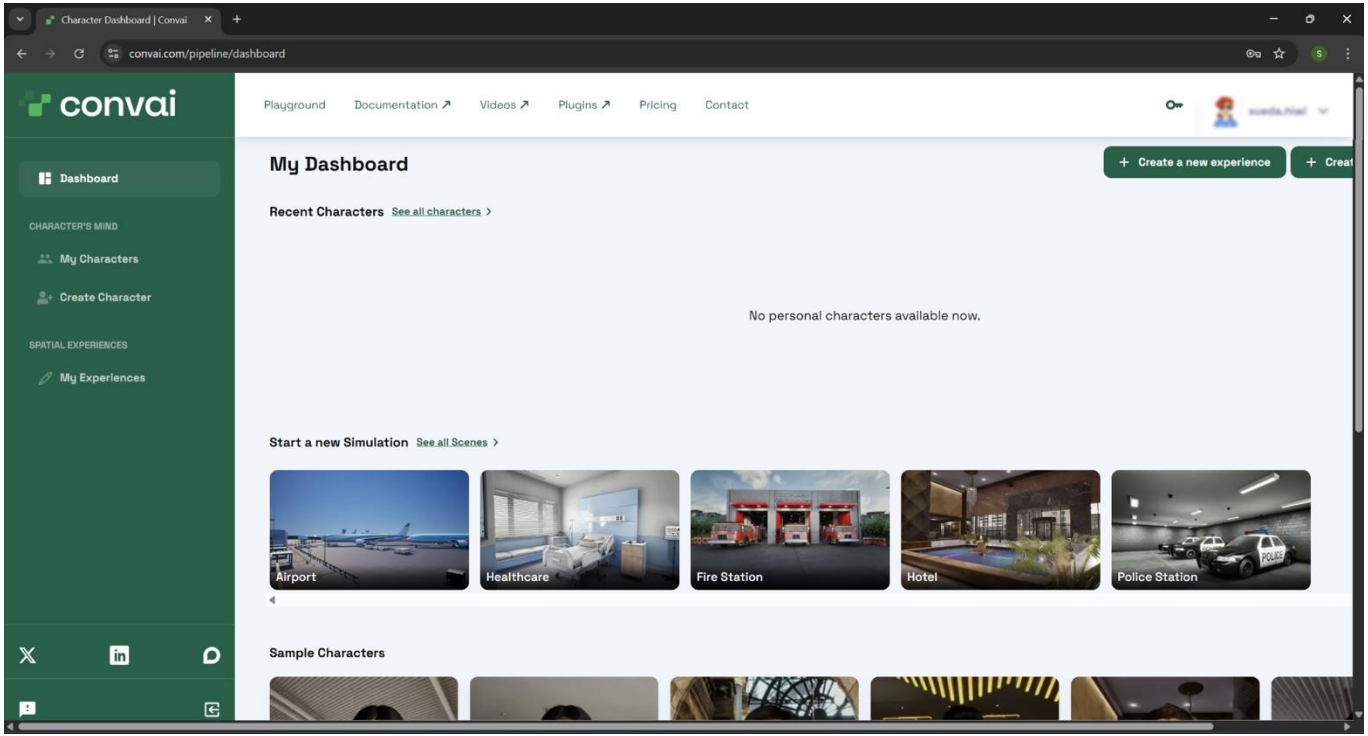
Once ready, we'll integrate this character into our Unreal Engine NPC using the **Convai Character ID**.
Let's dive in!

Sign up to Convai

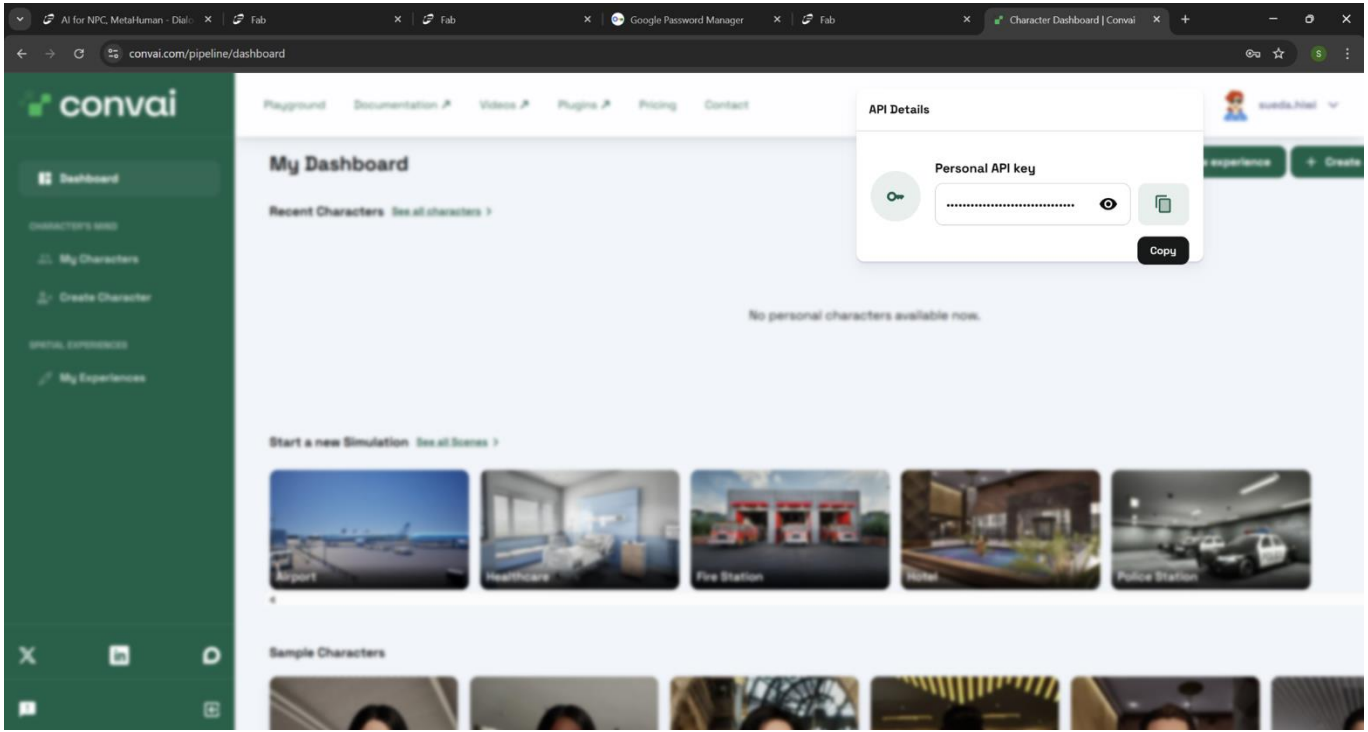
To begin using Convai, go to convai.com and **sign up** or **log in** to your account.
This is where you'll create and manage your AI-driven virtual characters.



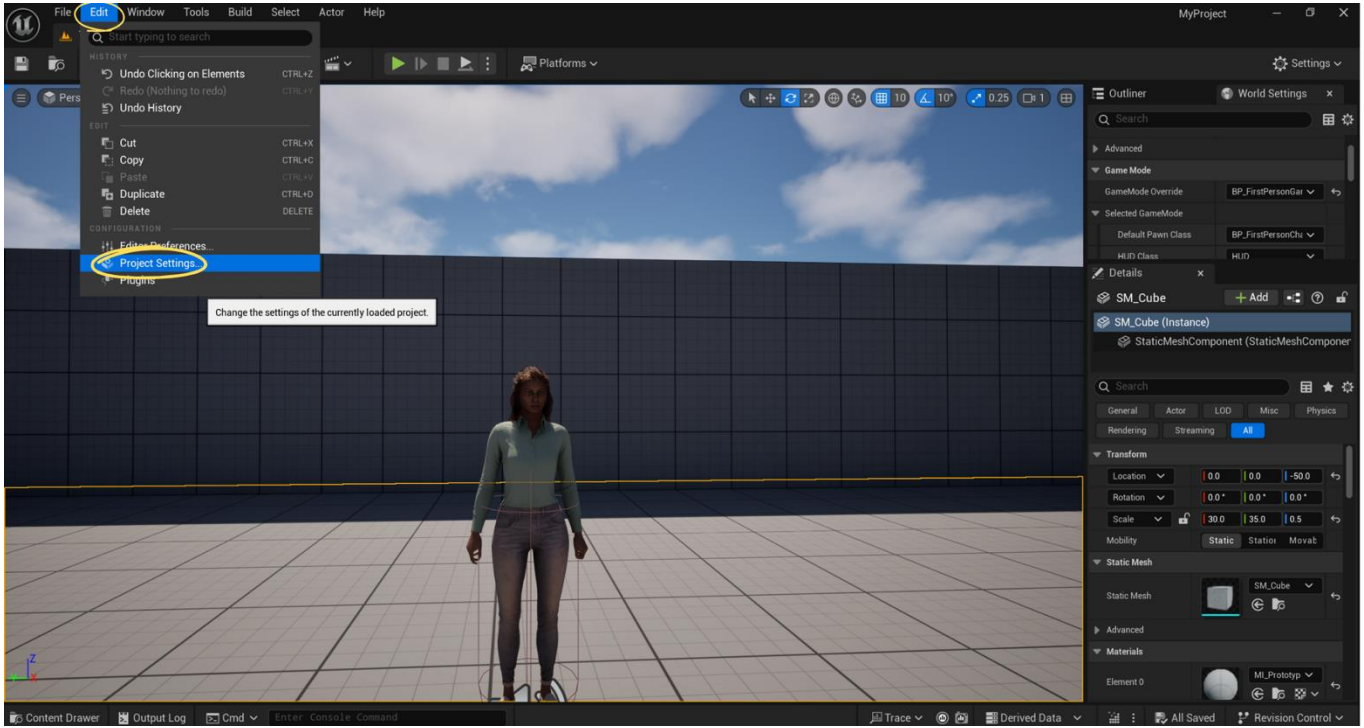
1. This is your main **dashboard** after signing in. You'll manage your characters and spatial experiences from here.



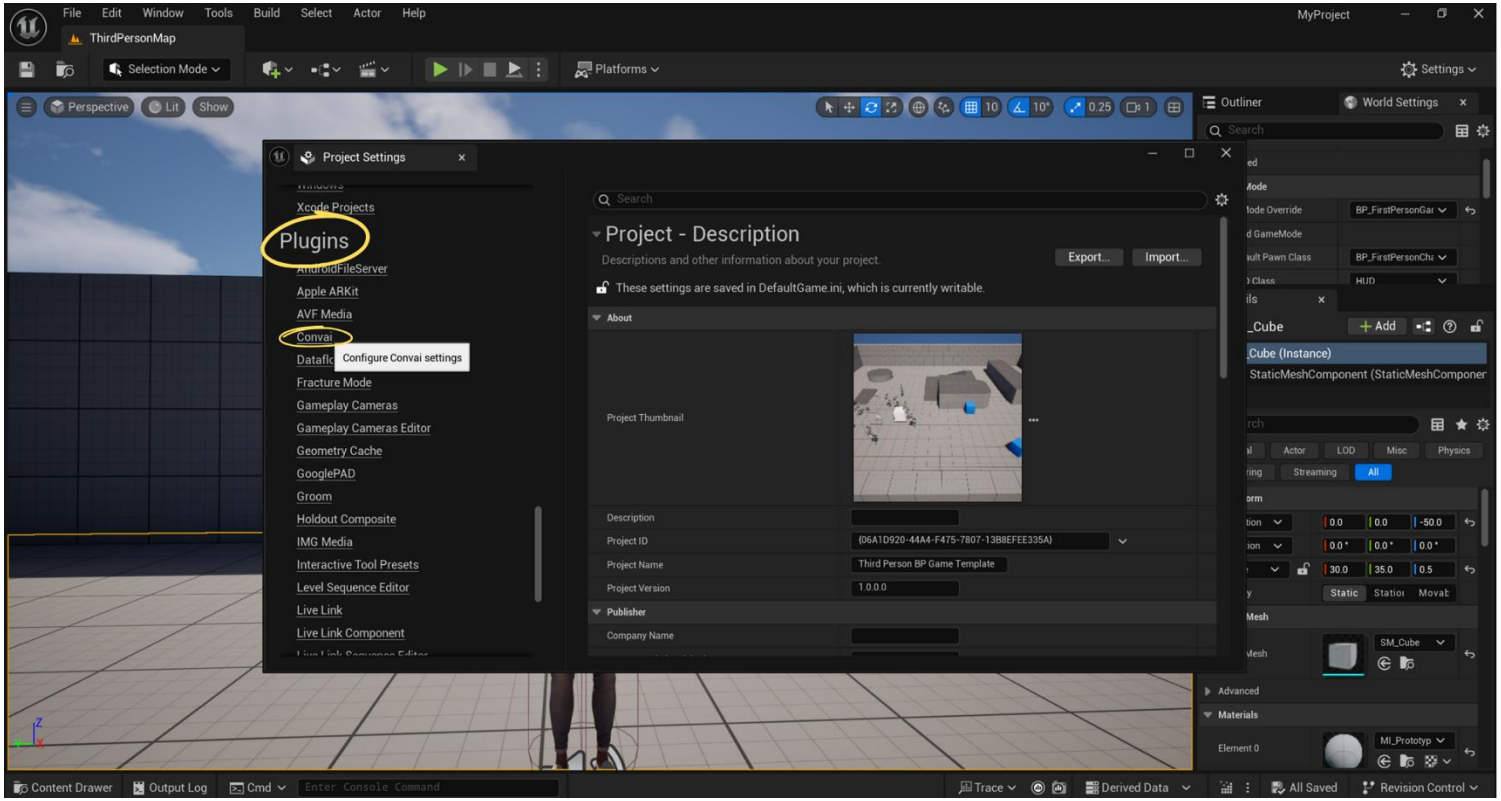
2. Click the little **key icon** in the top right to reveal your personal **API key** and **copy** it.



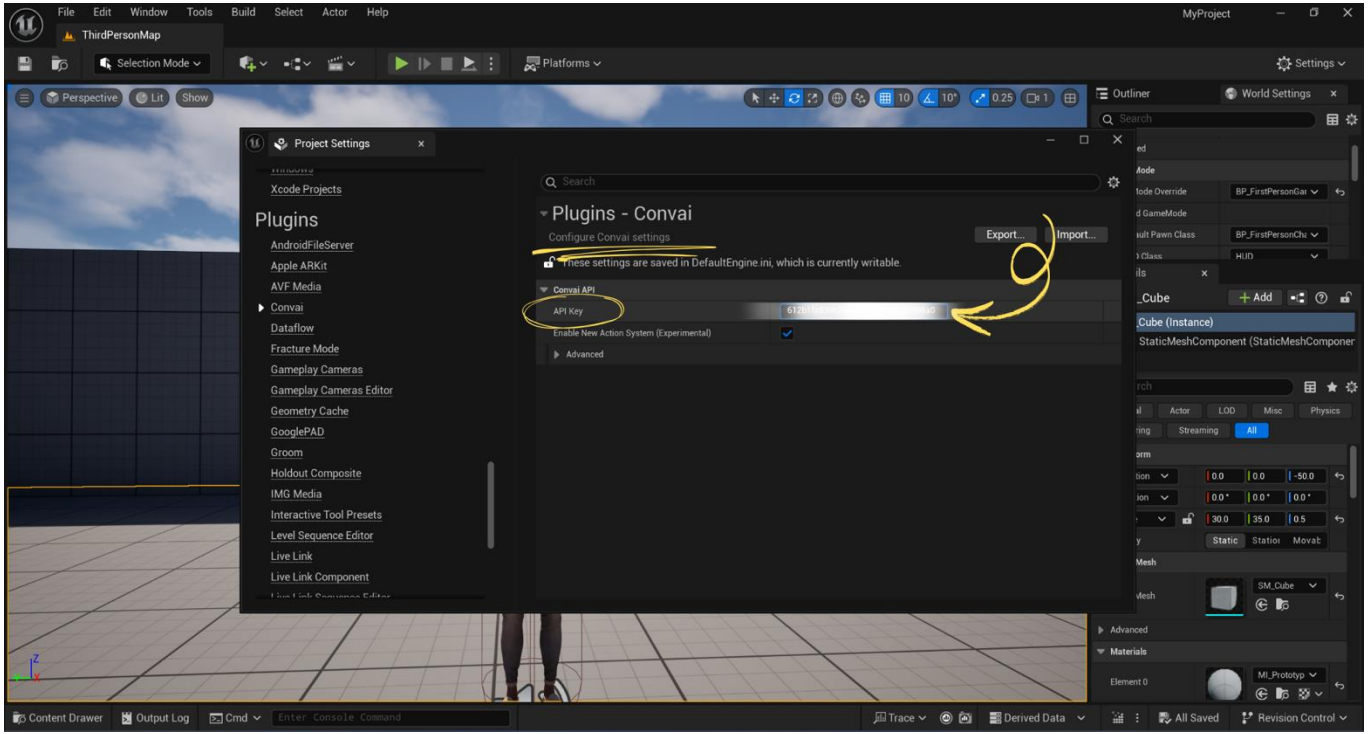
3. Go to **Edit > Project Settings** in Unreal Engine to start the plugin configuration.



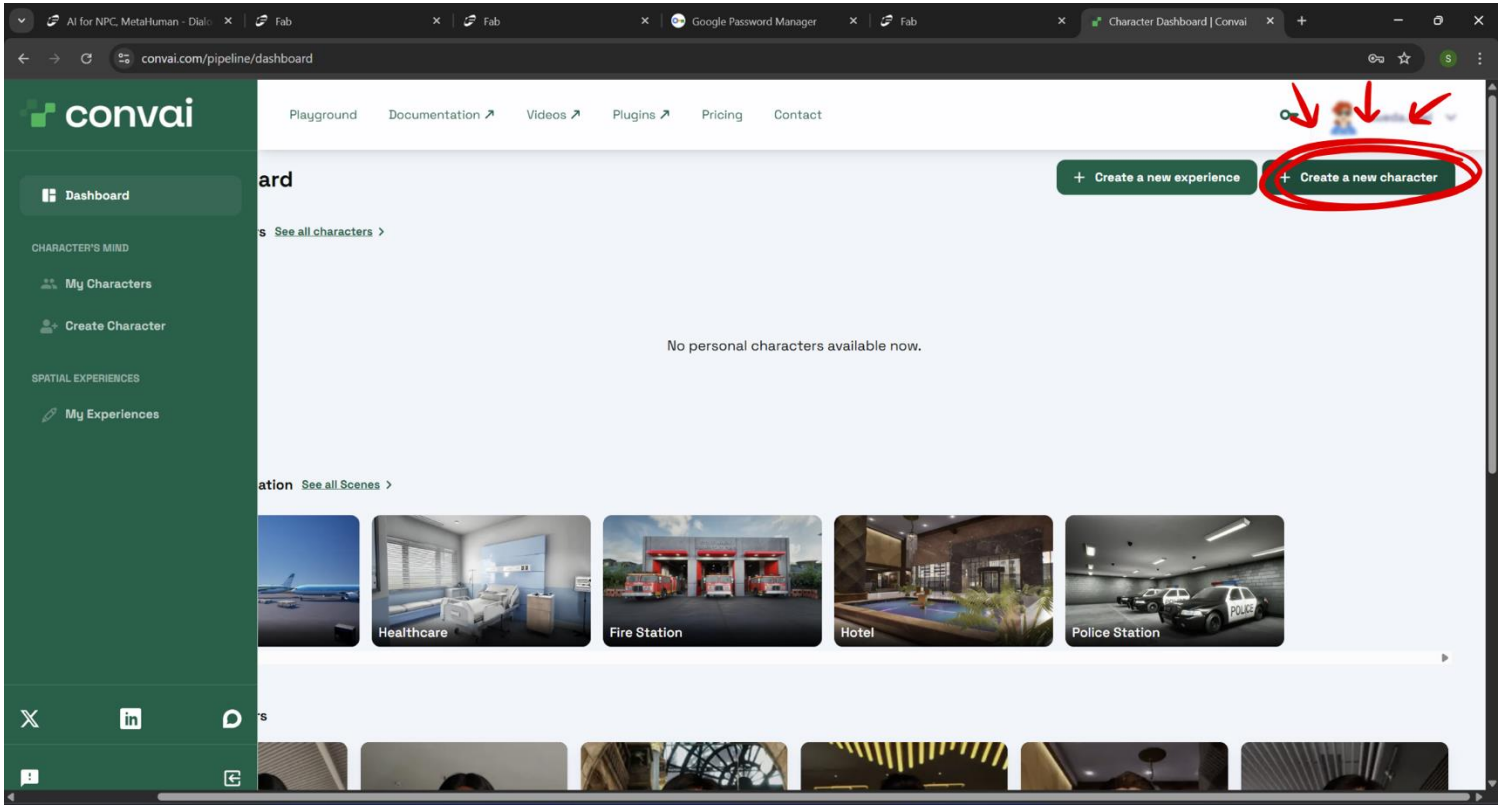
4. Scroll down the left panel and click **Convai** under the Plugins section.



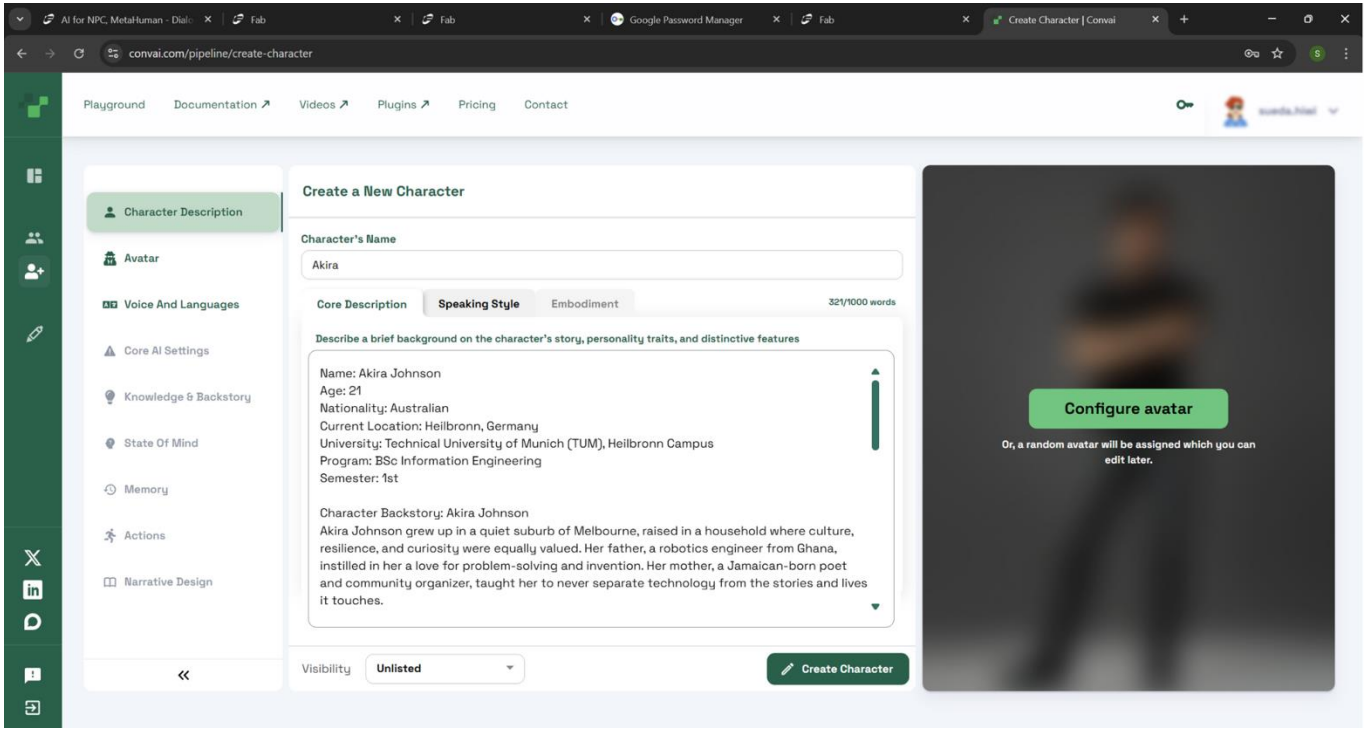
5. Paste your copied API key into the input field. Make sure to enable the **New Action System** checkbox as well.



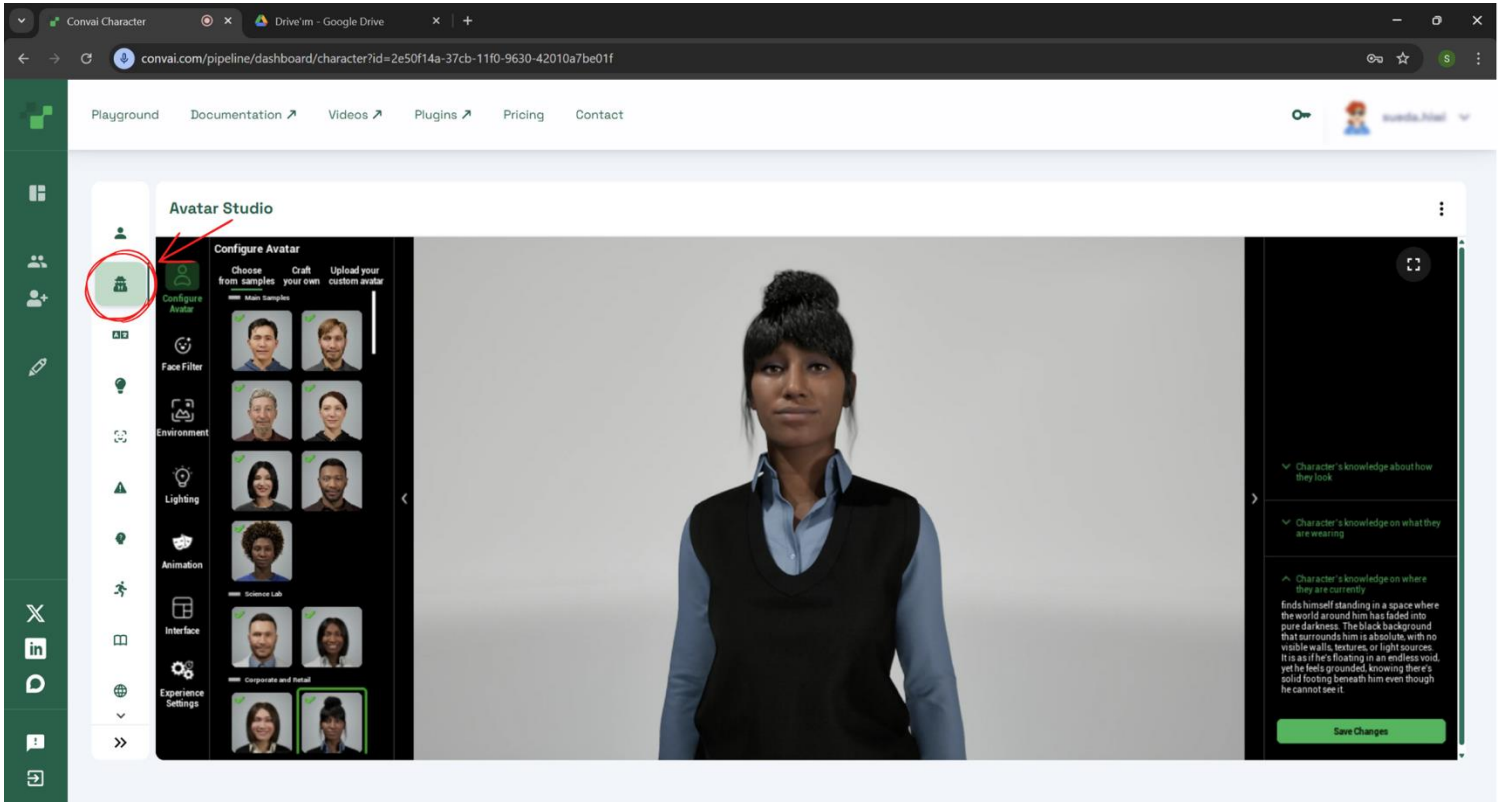
6. Back in Convai, click **Create Character** to start designing your NPC.



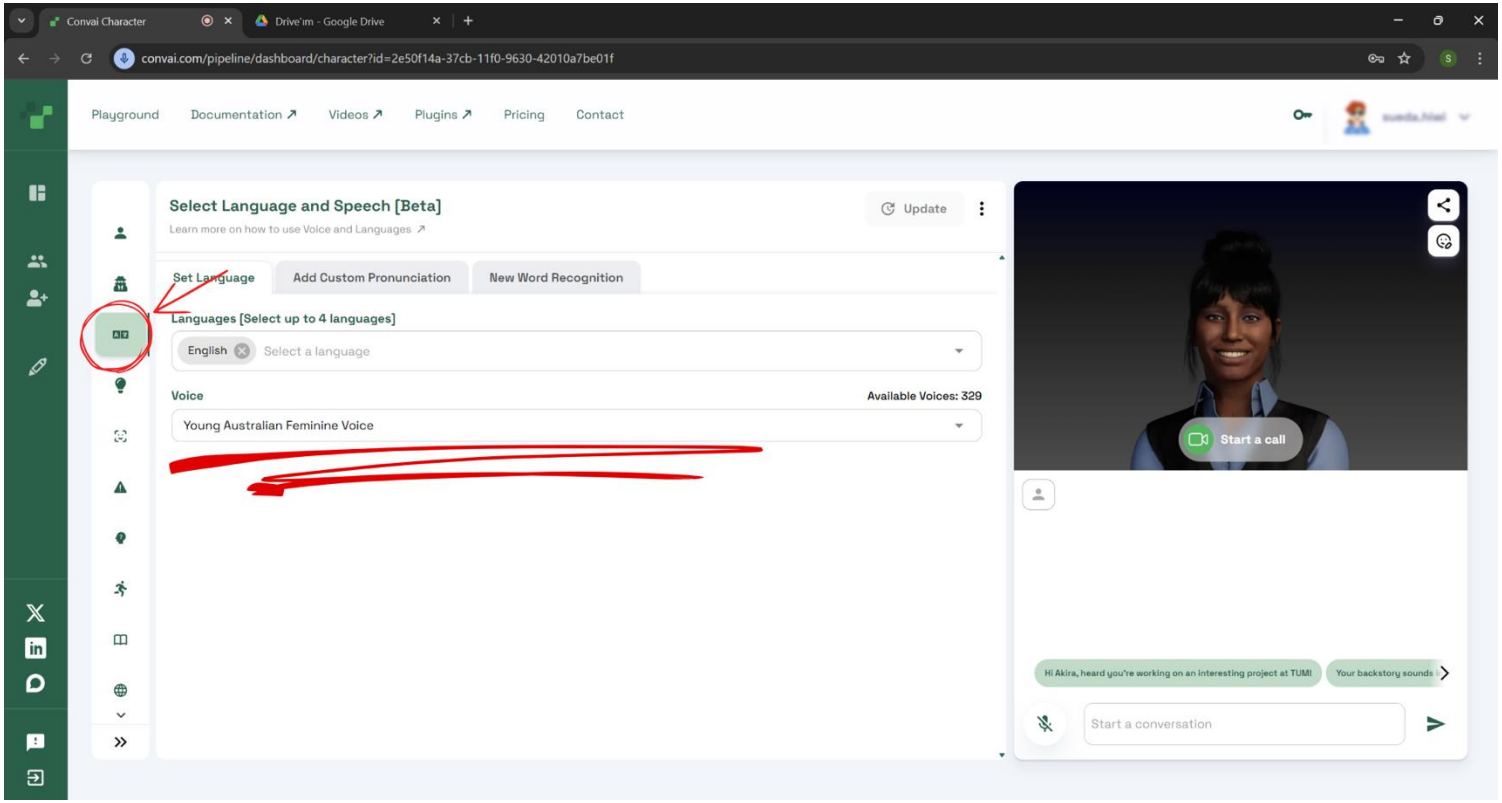
7. Here you'll define your character's **name** and a **background story**. (You can also generate it with AI.) Then, click on **Create Character**.



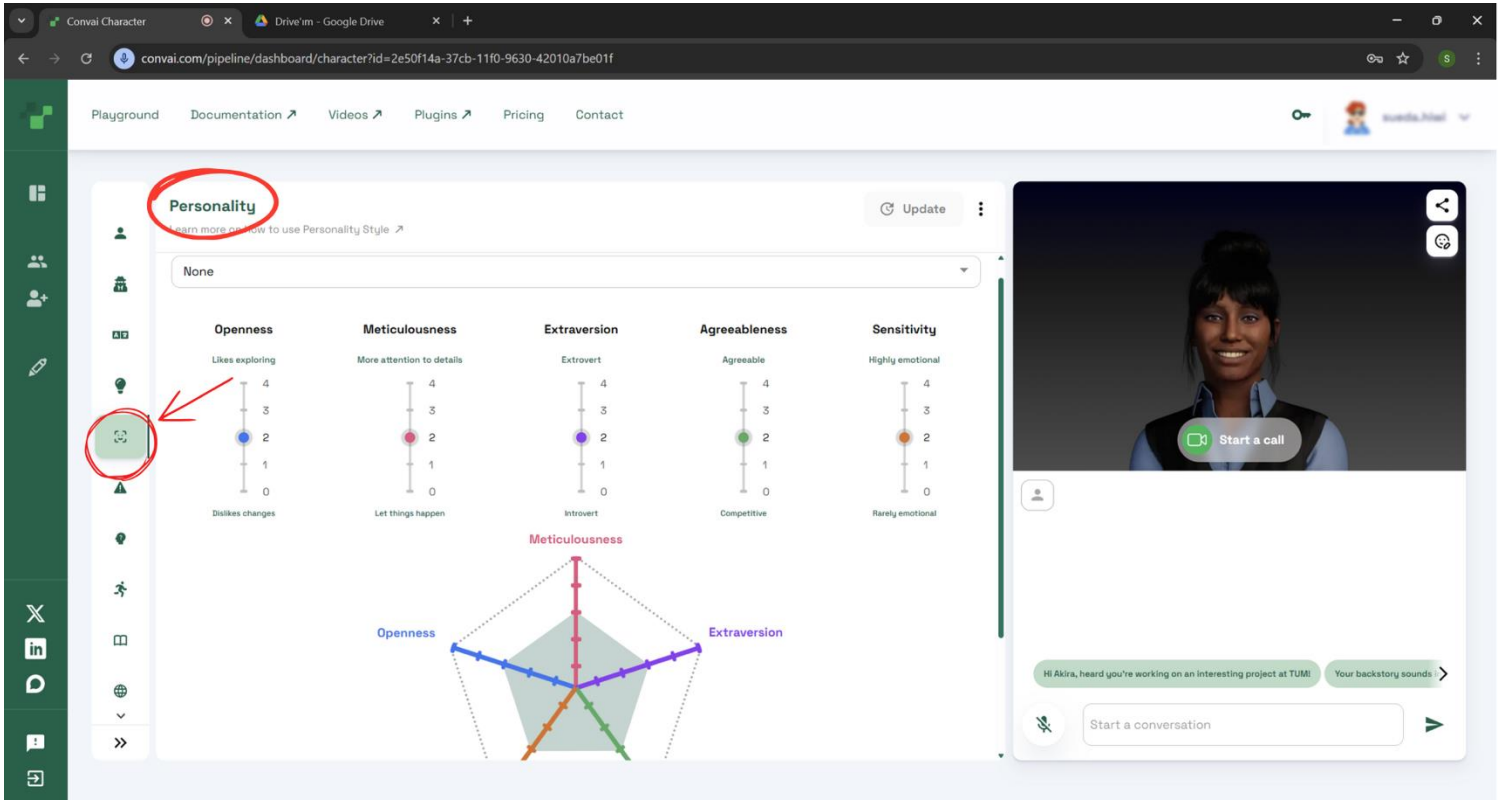
8. Then go to **Avatar Studio**, and choose one of them.



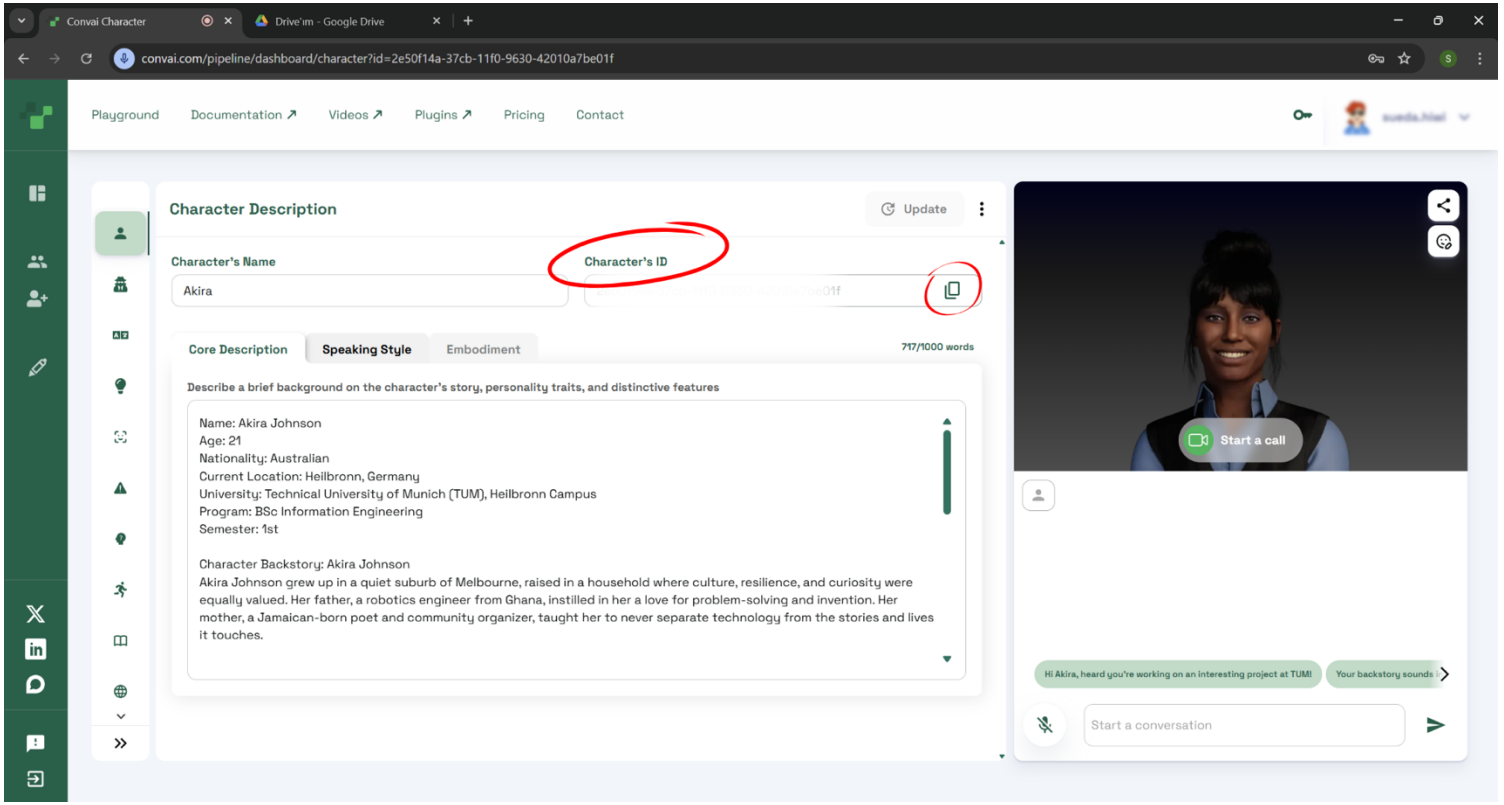
9. Here you can select **languages** and **voices** for the character.



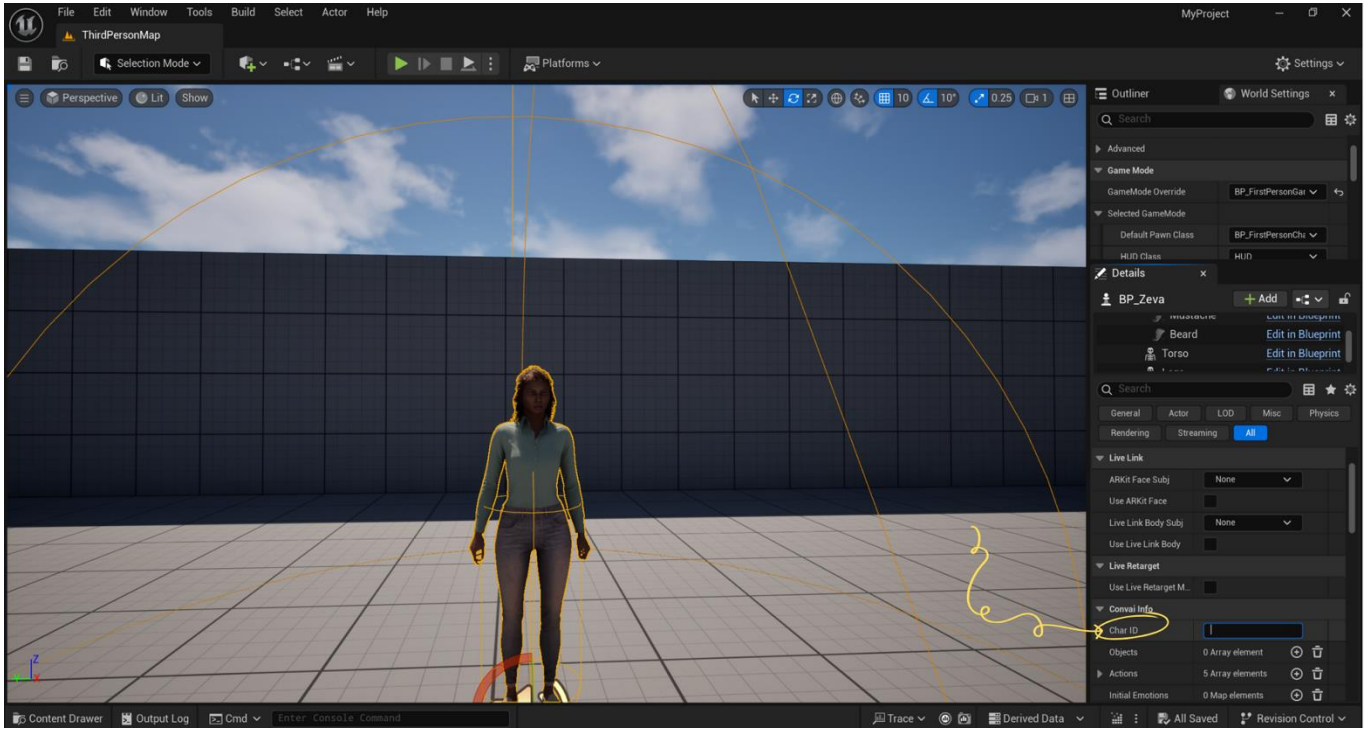
10. You can also change the **personality** settings of the NPC :).



11. Once you are done, go to Character Description > **Copy Character ID**.



12. Click on the character and go to Details > Convai Info > **Char ID** and paste it.



All done!

Once created, each character has a unique **Character ID**. This ID links the Convai character to your Metahuman inside Unreal.

Now we have linked them. Feel free to run and talk to your character and see how it speaks according to the created background story and other features!

Some Additional Settings for VR

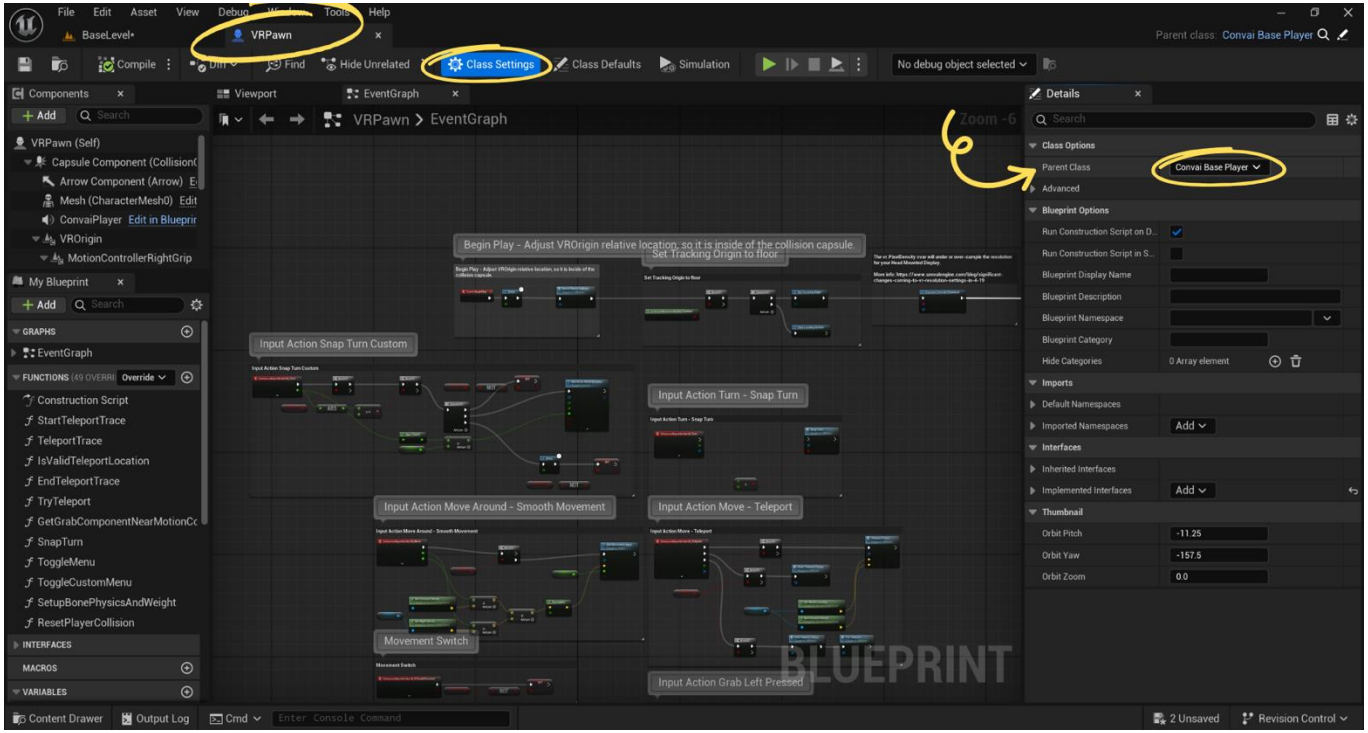
So far, we've successfully created an intelligent Metahuman powered by Convai and integrated it into a standard Unreal Engine scene.

But how does this change when we move into **Virtual Reality**?
This next section will adapt our LLM-NPC setup to support VR headsets and controllers, enabling real-time, immersive voice interactions in 3D space.

First, open an existing VR Project or create a new one, as already shown with Unreal Engine.

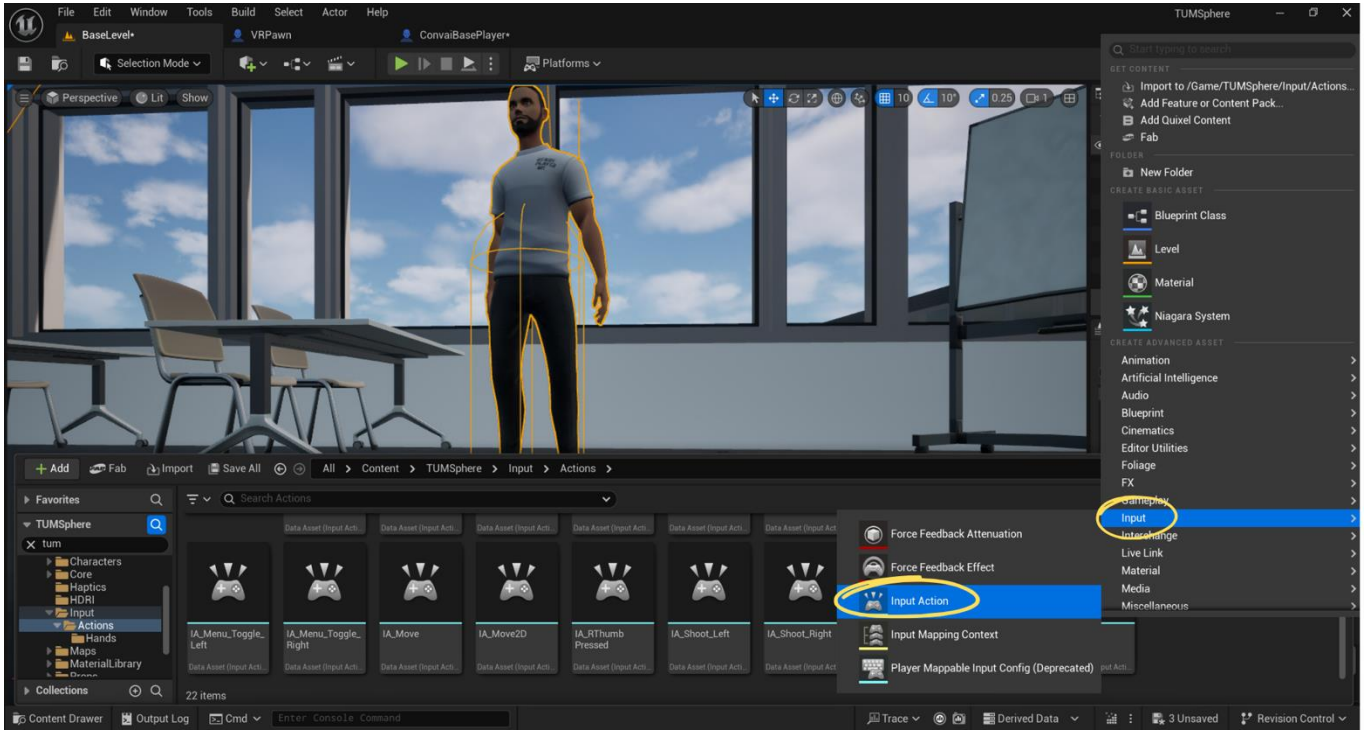
1. Go to **Content Browser** → Search VRPawn → Double-click

→ Open **Class Settings** → Set **Parent Class** to ConvaiBasePlayer



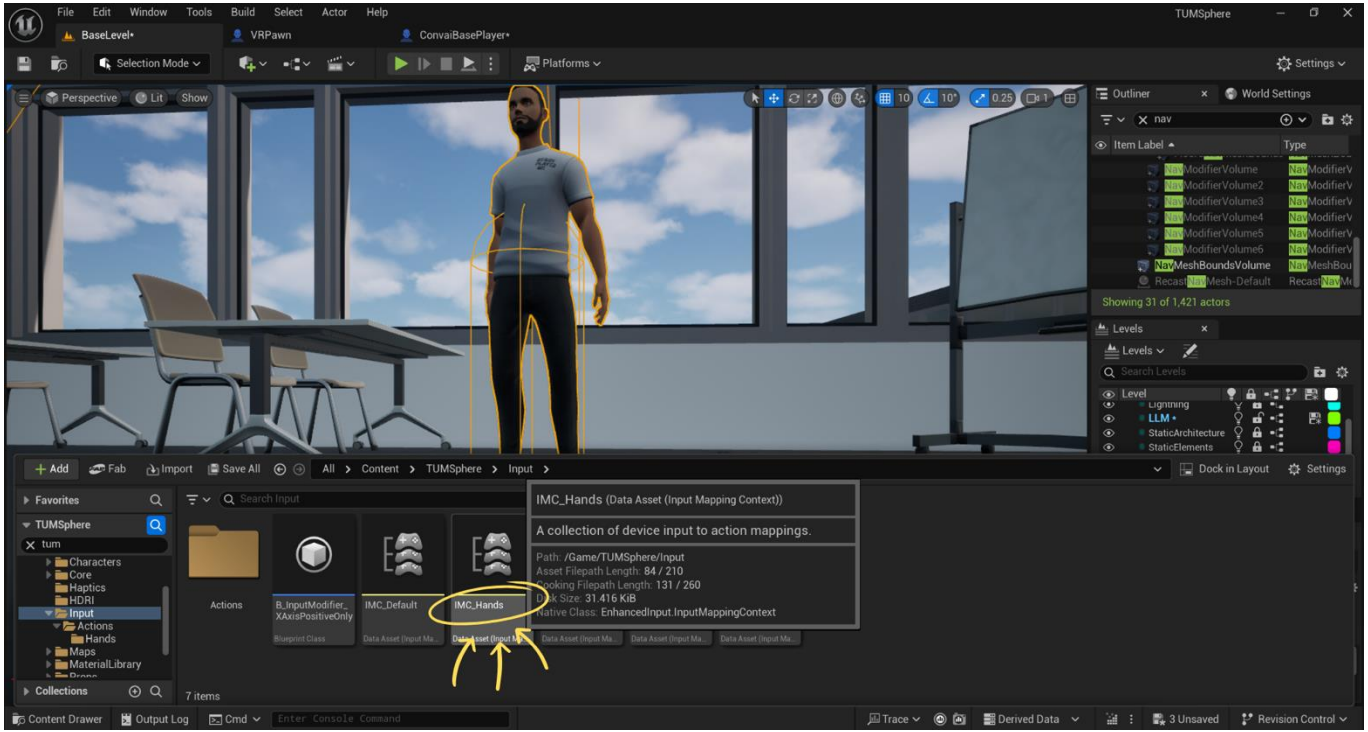
2. Navigate to: VR Template > Input > Actions

→ Right-click → **Input > Input Action** → Name it: **IA_Talk**

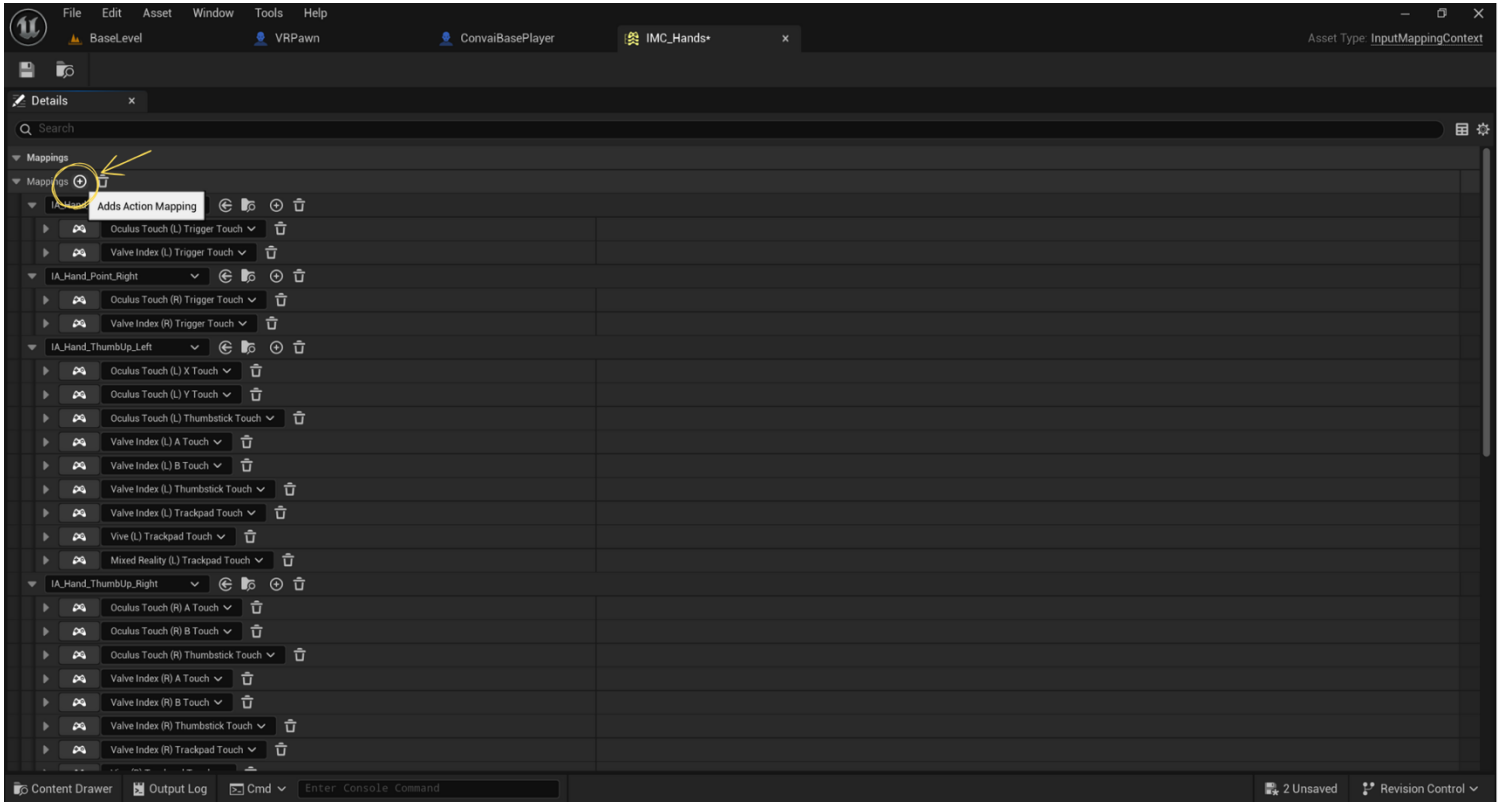


3. Go to: VR Template > Input > IMC_Hands

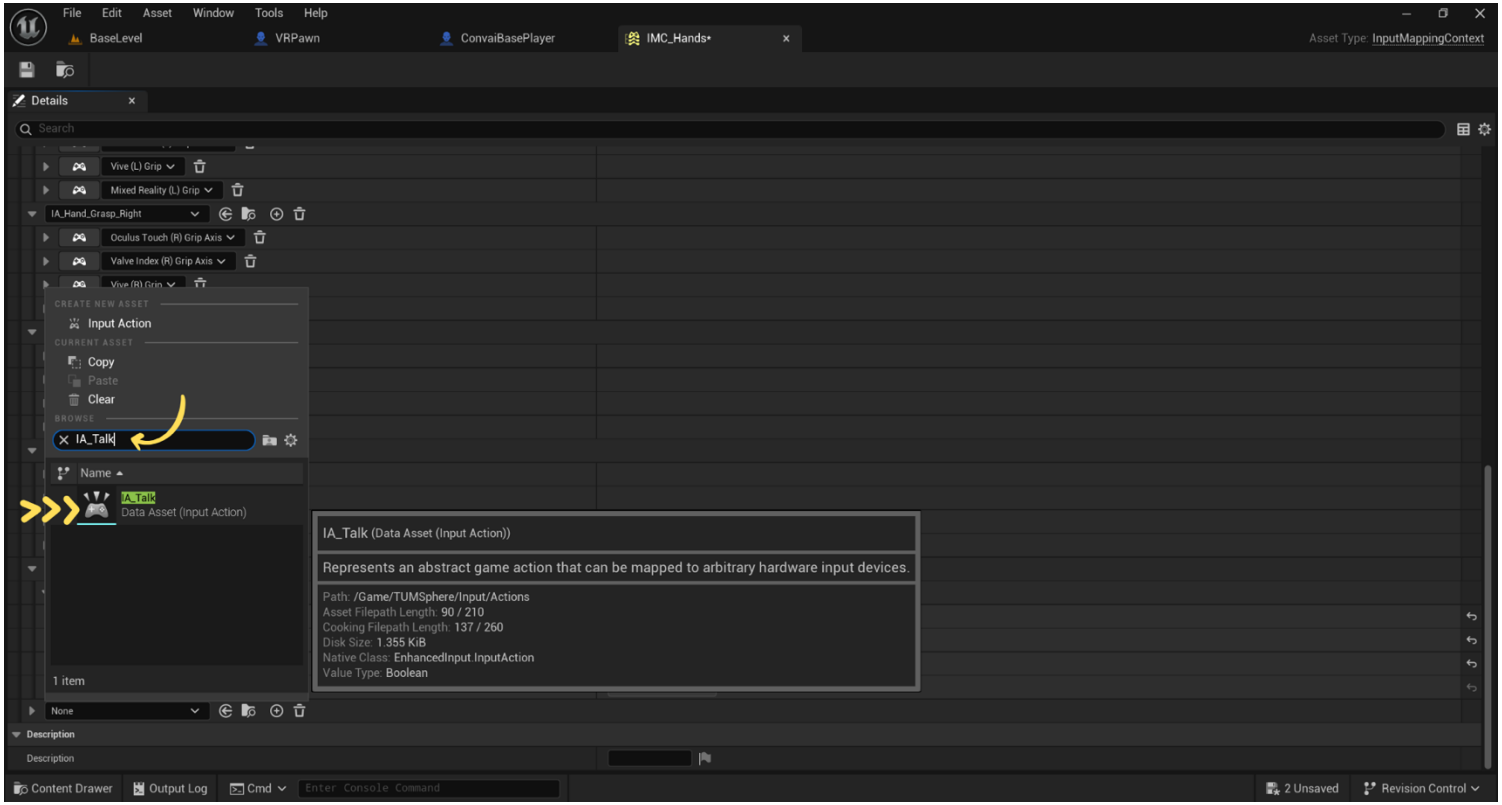
→ Double-click to open the mappings



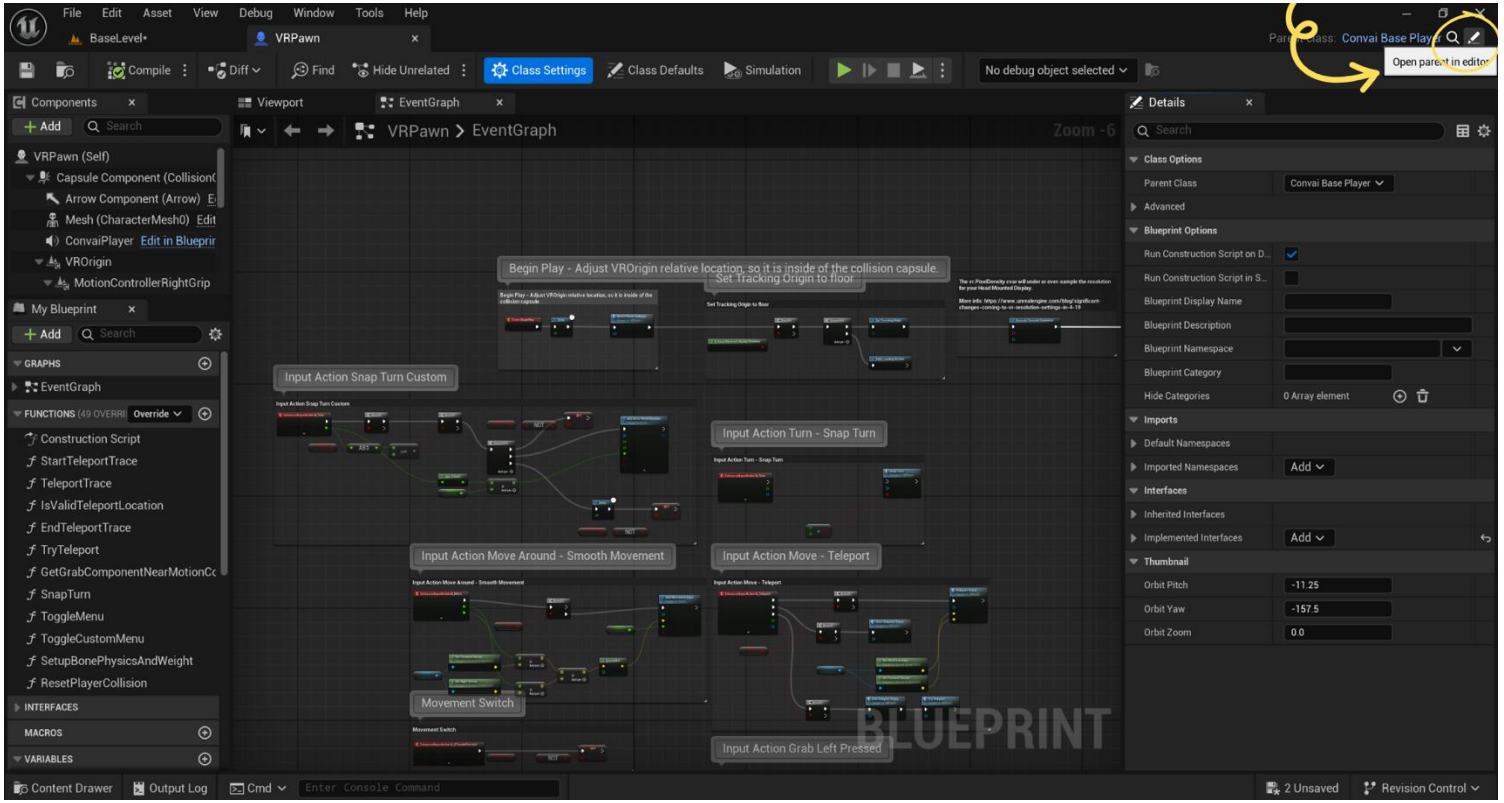
4. Click the + icon next to Mappings to add a new one.



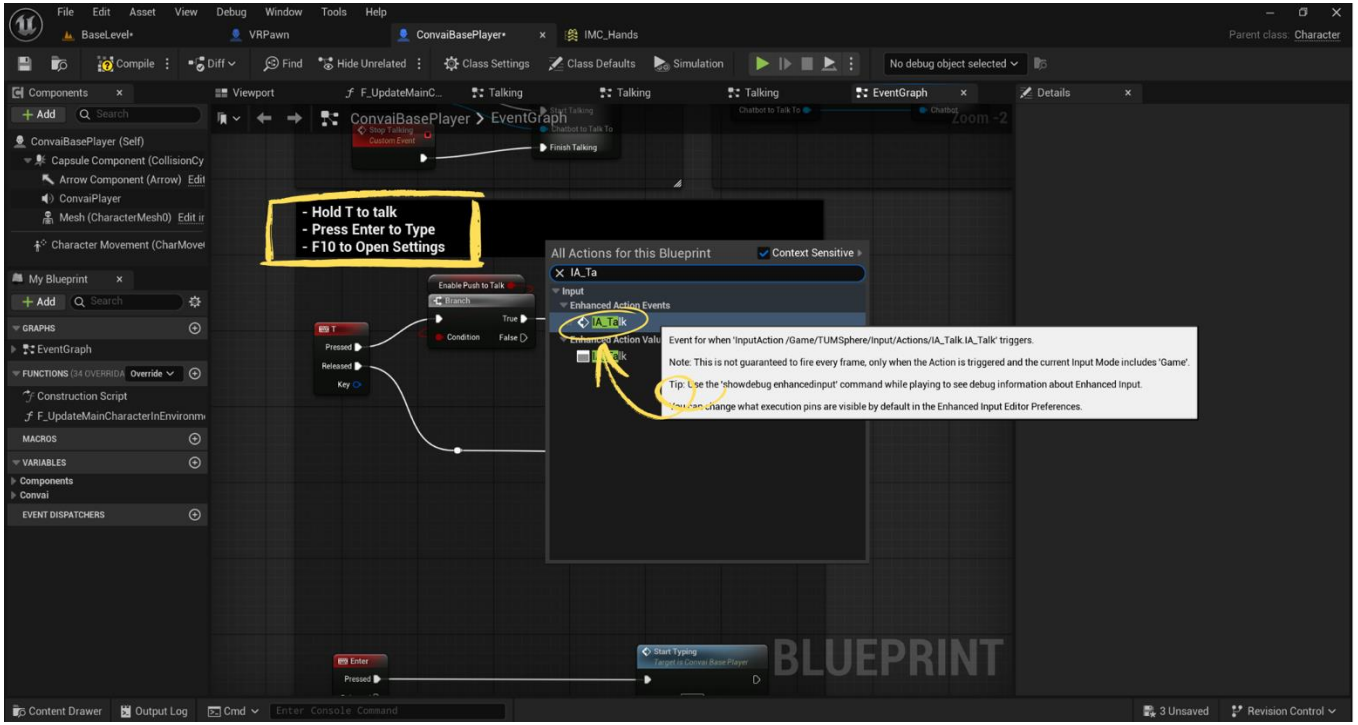
5. Select **IA_Talk** from the list and set input to **Oculus Touch A (Right) Press**



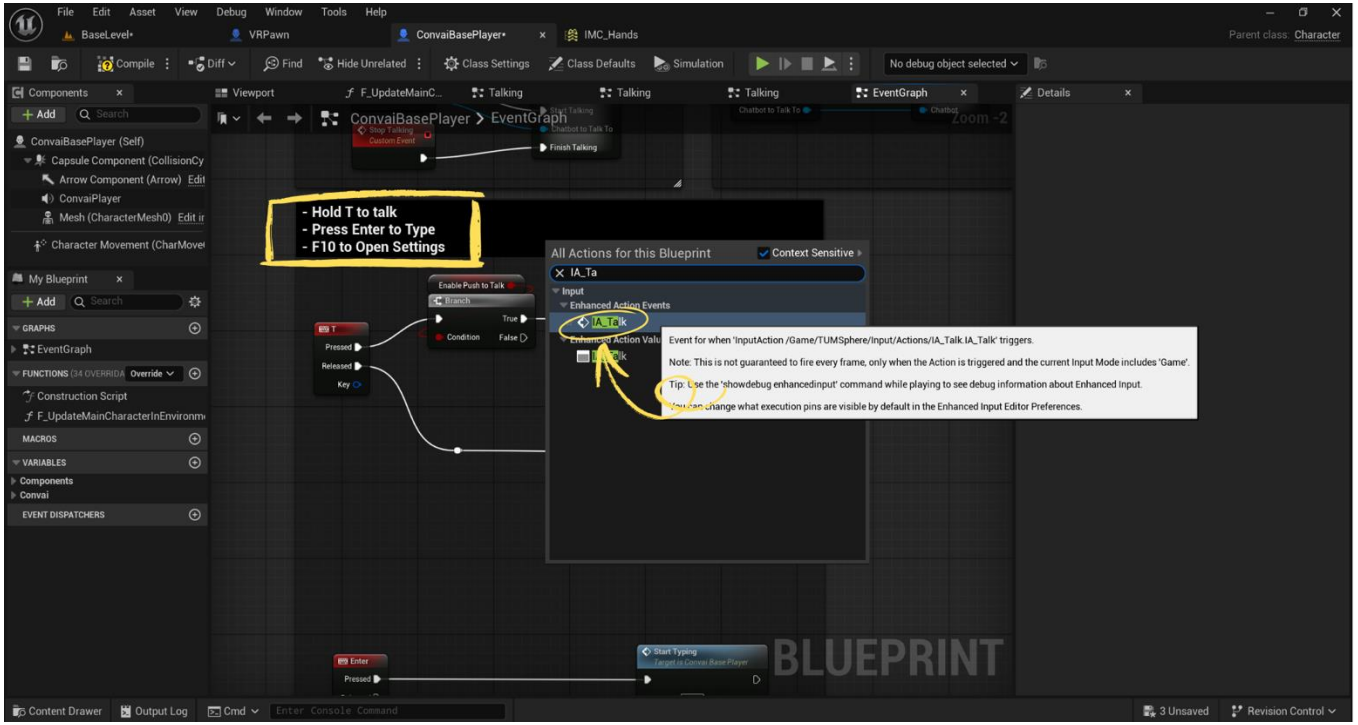
6. Go to **VRPawn** and click on the top right edit icon for **ConvaiBasePlayer**.



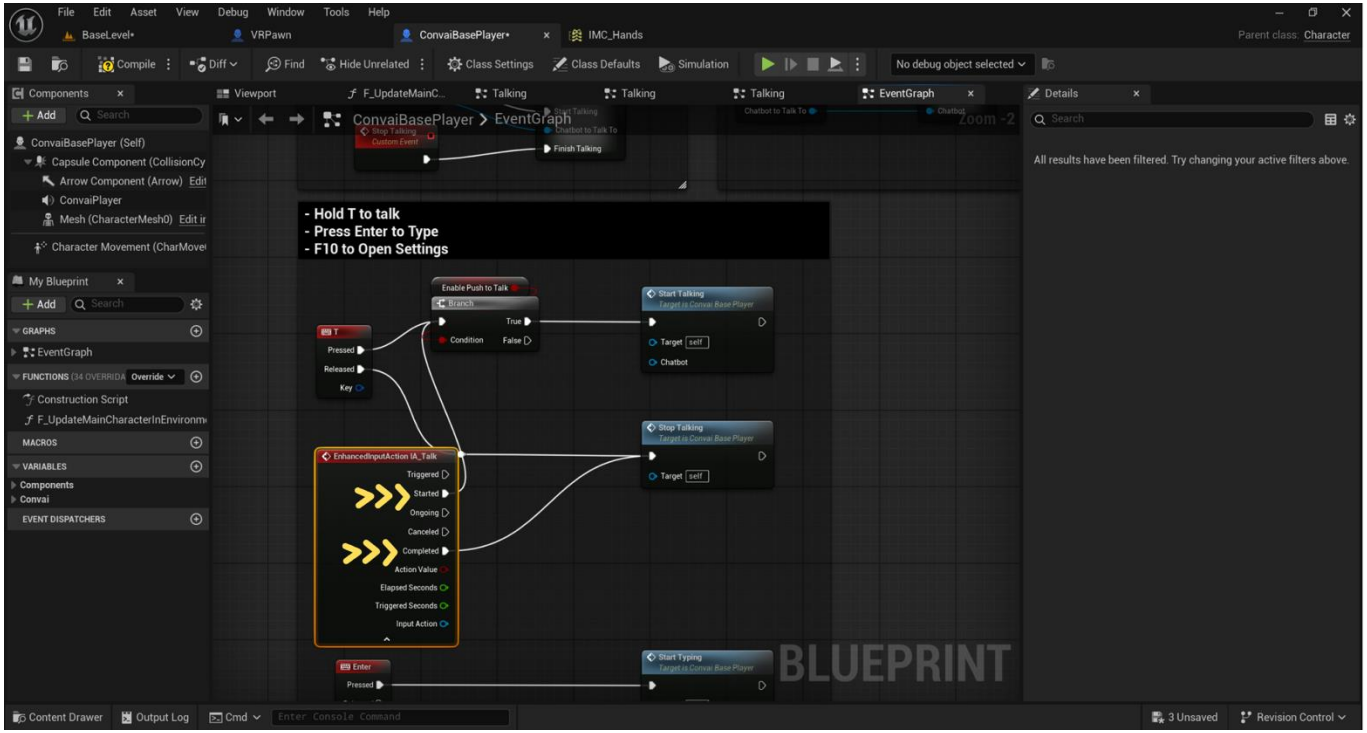
7. In the **Event Graph**, locate the section where character speech logic is handled (instructions shown in a comment box like "Hold T to talk").



8. Right-click in the graph and search for IA_Talk. Choose **IA_Talk** (Enhanced Action Event).



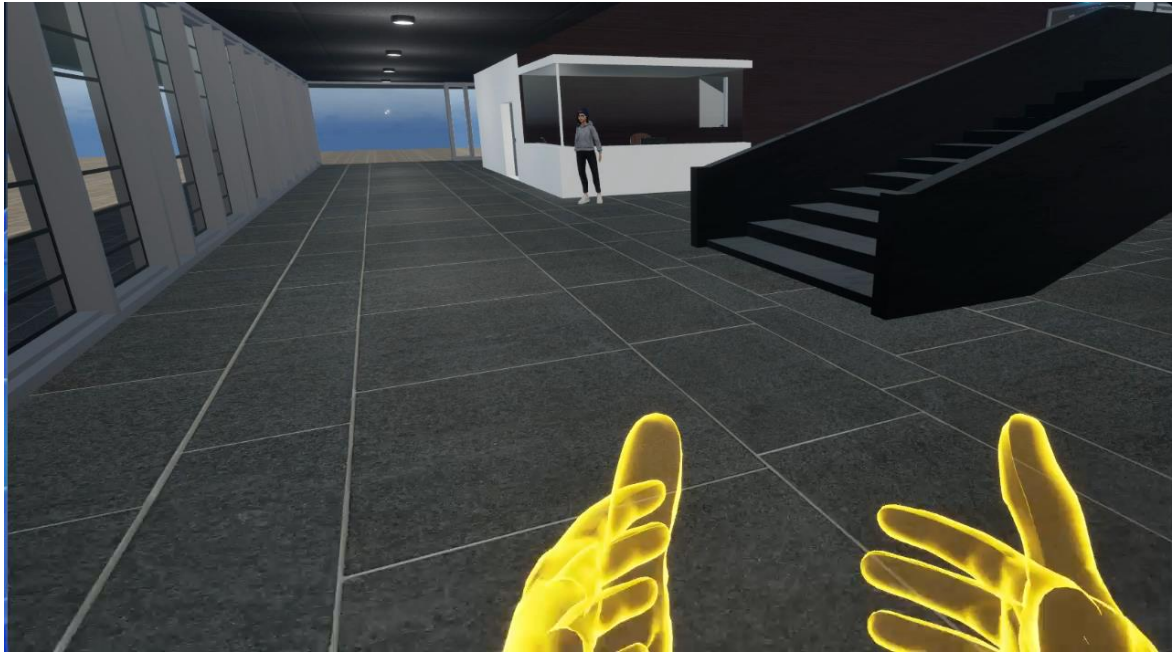
9. Connect: Started → Branch to enable Push-to-Talk → Start Talking Completed → StopTalking event.



✅ And that’s it!

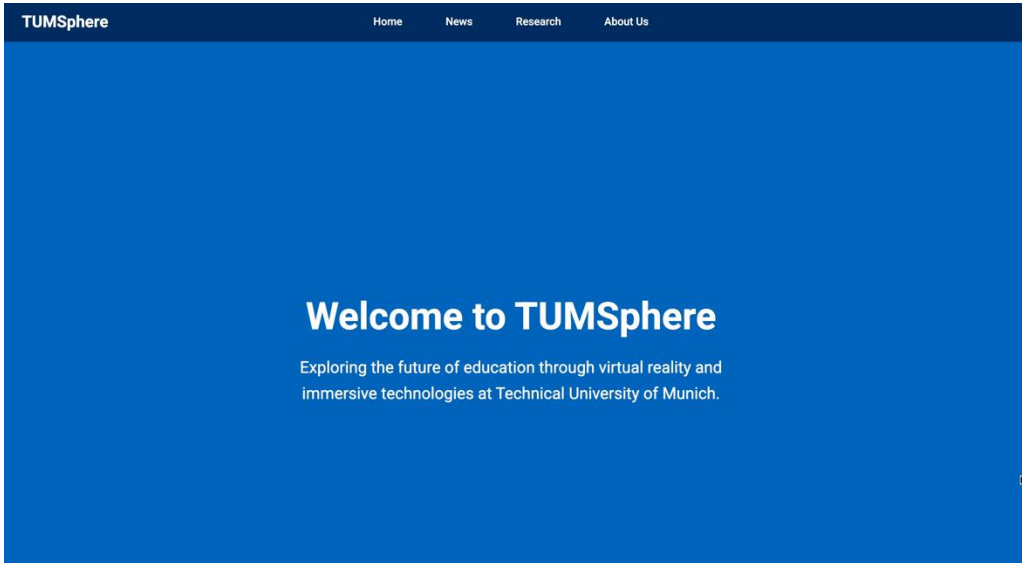
We have successfully integrated our intelligent AI-powered NPC into a VR environment , enabling live, immersive conversations with voice controls.

NPC Conversation Test



NPC Follow Interaction Test

TUMSphere



ICVR 2025
International Conference on Virtual Reality

TUMSphere

Thank you very much for your attention!
Any questions?

ICVR 2025
International Conference on Virtual Reality

Integrating AI with Meta Human Avatars in Unreal Engine

A complete hands-on guide to implementing smart NPCs in Unreal Engine 5.5 using Convai.

Dr. Santiago Berrezueta

Technical University of Munich

Germany

ICVR 2025
International Conference on Virtual Reality