Faculty of Engineering, Computing and Environment

School of Computer Science and Mathematics

Kingston University London

CI7800 Digital Media Final Project - Final Report

MSc GAME DEVELOPMENT (PROGRAMMING)

Leveraging Large Language Models for Dynamic Game Narratives: Become Human

Prajwal Shetty Vijaykumar - K2371155

Body of Creative Work
Email: k2371155@kingston.ac.uk

Supervisor: Prof. Vasileios Argyriou
Email: vasileios.argyriou@kingston.ac.uk

Release date: 10th January 2025

Table Of Contents:

Introduction and background: 3

Aims: 3

Objectives: 3

Literature Review: 4

Current State of the Art: 4

Inspirations: 5

Method: 5

Development Process: 6

Project Planning and Design: 6

LLM Integration: 6

Entity Component System In Unreal: 7

Prompting LLMs: 7

Level Design and Environment: 8

Subsystems and Runtime: 9

Testing and Optimisation: 10

Tools and Technologies 10

Project Structure: 10

Outcomes: 11

Cost breakdown: 11

Key Scenarios and Player Feedback: 12

Ethics: Social, Legal, Data Security: 12

Legal and Copyright Issues: 12

Ethical Considerations: 13

Discussion and conclusions: 13

Strengths: 13

Limitations: 14

Future Work: 15

References: 15

Introduction and background:

The modern-day story-driven, role-playing games have evolved into complex, sprawling franchises. While these carefully crafted experiences often result in highly satisfying outcomes for players, as seen in titles like Uncharted (PlayStation, n.d.), and Red Dead Redemption (Games, n.d.), they also come with some drawbacks. Firstly, the development of such games is an incredibly time-consuming and labor-intensive process. As game scopes continue to expand, development teams face tremendous pressure to create ever-larger and more immersive experiences, pushing the boundaries of what's feasible within the current production scale. Secondly, despite the substantial scale of these games, the overall outcome always stays consistent across the millions of players who buy these games. While this consistency allows developers to maintain control over the gameplay experience, it can also limit the potential for more diverse and personalized adventures. With the rise of large language models (LLMs), there is an opportunity to revolutionize the whole development process. These systems have reached a level of sophistication where they can understand and respond to many of the natural human languages, including English, with near real-time response speed. The generated replies are also contextually aware and smart, opening up new possibilities for player-game world interactions. By leveraging LLMs, we can begin to break free from the constraints of predefined paths in game design. Players can now experience dynamic game flow and NPC interactions, potentially leading to storylines and endings that even the game developers themselves couldn't have anticipated.

Aims:

The primary aim of this project is to develop a proof-of-concept game that showcases the potential of LLM integration for dynamic narrative and gameplay experiences. The concept of "dynamic narration" involves crafting a gameplay experience where the story progression, player interactions, and NPC behavior adapt fluidly to the player's choices and investigative path. While maintaining a fixed urban environment as the game's backdrop, the project emphasizes the use of AI-driven dialogue and decision-making systems to enhance replayability and engagement. Additionally, the project aims to explore the feasibility and impact of simulating a busy urban environment with numerous LLM-driven NPC agents, enabling crowd dynamics and examining how player interactions with them contribute to narrative development.

Objectives:

1. Dynamic Narrative Integration: Setting up an LLM-driven narrative system to set the overall mission context and eventual progress.
2. Crowd Simulation: Simulate a bustling urban crowd using LLM-driven NPC agents each with unique life stories, purpose, and behaviors.
3. Dialogue and Interaction System: Create a dialogue system for players powered by LLMs, allowing for natural, context-aware interactions with individual NPCs.
4. Game Environment and Spatial Awareness: Develop a compact, urban environment with distinct landmarks (e.g., fast food chain, cinema, park) that serve as narrative touchpoints allowing the agents to plot stories around them.

5. Replayability and Scalability: Focusing on a replayable experience making each gameplay session unique. Develop a modular, scalable LLM and Code integration architecture that can eventually scale to bigger projects.

Literature Review:

The field of AI and game development is evolving at an unprecedented pace, with new approaches and tools emerging almost daily. Large Language Models (LLMs) like GPT-4 and Claude are improving rapidly, enabling capabilities that were unimaginable just a few years ago. These advancements have opened up exciting possibilities for creating dynamic, adaptive, and immersive storytelling experiences in games. The quick evolution of LLMs allows for real-time, context-aware interactions, making the line between scripted and emergent narratives increasingly blurry.

Current State of the Art:

The use of Artificial Intelligence (AI) and Role-Playing Video Games(RPGs) has been increasing interest in recent times, especially by improving how Non-Player Character NPCs interact with players. Existing research has shown how large language models (LLMs), like ChatGPT, can create dynamic and context-aware dialogues for NPCs that adapt to the actions, choices, and environments within a game (Csepregi, 2021). The study focuses on context-aware NPC dialogues but misses how they can help create a connected and consistent story, which can lead to disconnected storytelling. In addition, Gallotta et al. (2024) talk about how new LLM technologies are changing game design. These include things like autonomous players, random content generation, and dynamic game mastering. These models can help create realistic dialogues and maintain the feel of the game by making NPCs act like real people. However, they still face challenges like memory issues and sometimes giving wrong or inconsistent information. A Survey on Large Language Model-Based Game Agents (Hu et al., 2024) identifies significant advancements in the architecture of LLM-based game agents, including perception, memory, and reasoning modules. However, the report mostly focuses on how to improve the gameplay side of things, instead of continuous storytelling. While it talks about grounding LLMs into in-game environments, they don’t focus on how NPCs can adapt to long-term player choices, which stays underexplored. This report aims to fill that gap by proposing methods to ensure NPC interactions dynamically adapt to player choices and keep the story consistent throughout long gameplay sessions, creating a more engaging experience.

The report also builds on the ideas explored in SceneCraft (Kumaran et al., 2023) where the agentic dialogues were attempted to make dynamic by giving contexts in terms of the generated scenes. However, it does not fully explore the potential of dynamic narration as its focus lies on conversations. There is also a project that explores players and LLMs working together to solve quests (Rao et al., 2024), the paper explores how the LLM-NPCs can help the player to make the game more immersive and also help them solve the game faster in Minecraft (Mojang, 2011). Kalbiyev (2022) investigated the effectiveness of LLM-generated dialogues by comparing them to actual human-generated dialogue datasets from Fallout 4 (Bethesda. n.d.). This was 2022 so the project was built upon fine-tuned GPT-2. Another paper that researches script generation instead of dialogues is The Turing Quest (Chen Gao and Emami, 2023), it demonstrates that their pipeline, with GPT-3, generates NPC scripts that can successfully deceive judges into believing they were written by humans. Now with even more advanced reasoning and large language models, the project aims to achieve even more realistic conversations between the player and NPCs while also impacting the overall narrative.

Inspirations:

The project's title "Become Human" is taken from the game "Detroit: Become Human" (Quantic Dream, 2018) as the project's initial inspiration came from it. The game, although was from pre-LLM and ChatGPT times, explored dynamic narration to a great extent, even today it is one of the best examples of giving the player "choices" and how their choices can change the story of the game, the game itself had a lot of endings which heavily depended on player choices throughout the gameplay. The movies The Matrix Resurrections (Warnerbros.com, 2024) and Free Guy (20thcenturystudios, n.d.) also played a role in conceptualizing the project, as they both explore human-like NPCs who eventually gain consciousness.

Method:

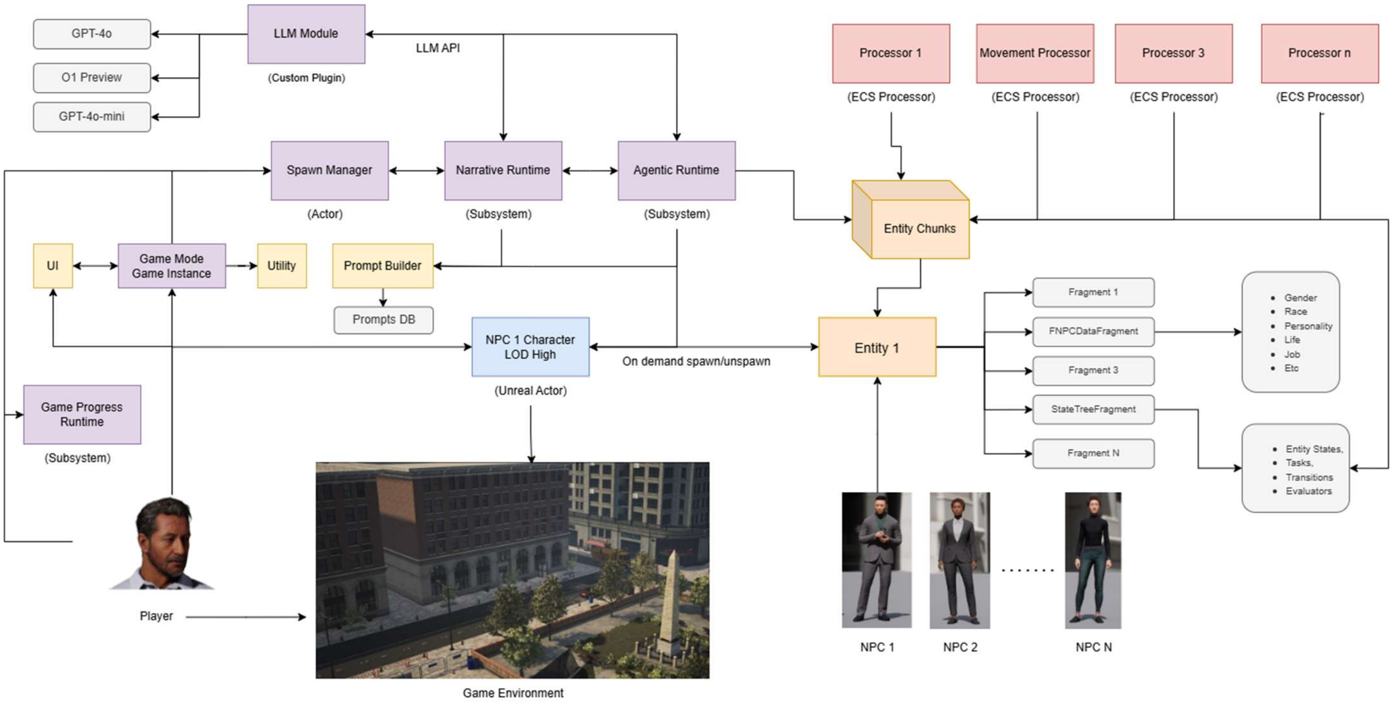


Figure 1: A simplified architecture diagram of the project

The development of the project followed an iterative and agile methodology. The primary focus was to integrate LLM-driven capabilities with Unreal Engine's ECS features to create a dynamic and interactive detective game. The methodology combined testing, iterative refinement, and modular design to ensure scalability and flexibility in the system. The development was structured into bi-weekly sprints, over the course of the initial three months, and then an additional two-week sprint towards the end of the deadline. The Project was built using Unreal Engine 5.3 with most of its components written in Unreal C++, it leverages the available ECS plugins called MassEntity, MassGameplay, and uses OpenAI GPT APIs to process LLM jobs. It also includes subsystems and actor runtimes to generate dynamic prompts, process LLM responses, and execute these interactions seamlessly within the game level, ensuring a high degree of context awareness and narrative coherence (Figure 1).

Development Process:

Project Planning and Design:

One of the first things the project aimed to achieve was picking a foundational LLM API for the game, so most of the research involved comparing the output qualities across various ranges of models, this was before reasoning models like O1, O3, and Deepseek, so GPT-4o was picked as a suitable model for the project as its narrative and dialogue responses were satisfactory, especially after giving some context about the project, the model was also well priced and OpenAI also has one of the most reliable LLM APIs among the bunch. Once the reasoning models were released on a later date, the project also attempted to integrate them but the latency of response time with the O1 preview models made sure that they weren't the best models for real-time gameplay scenarios, at least at the current state.

Another aspect of planning and design went into picking the right kind of environment and level for the project and to set the overall scope and complexity. Initially the plan was to integrate a big city chunk from the Epic's City Sample Project as the level for crowd spawning and interactions but it was quickly dropped after evaluating the complexity that comes with large levels. The project now has a small custom city block with multiple points of interest.

LLM Integration:

Multiple tools for LLM Integration were explored in Github and Unreal Marketplace but although none of them satisfied the requirements the project did well. It required a plugin that can support multiple LLMs from different organizations, especially during the testing phases, and also it should be lightweight as it didn't require multiple other API features like image generation, etc. So a custom Unreal Plugin was developed for the project called "UnrealGenAISupport" and iteratively added more API features as the requirement changed, it now supports multiple OpenAI models and has support for Chat completion and Structured Output features, which is a way to ask LLMs to respond in a JSON format based on the given Schema.

Since the LLM apis was set up as an entirely different scope it helped massively in debugging any issues with either LLM responses or its handling on the project side. The initial LLM outputs were indeed problematic in terms of coherence as they used to just generate good made-up stories that couldn't be traced back to the environment. The initial hunch was that the LLMs of the time were not good enough to understand the prompts well to perfectly act like an NPC, but eventually, it was pinpointed to lack of contextual information in the prompting which resulted in them picking up whatever ideal case response they could make up. So later on in the project, a more complex prompt system was set up.

Entity Component System In Unreal:

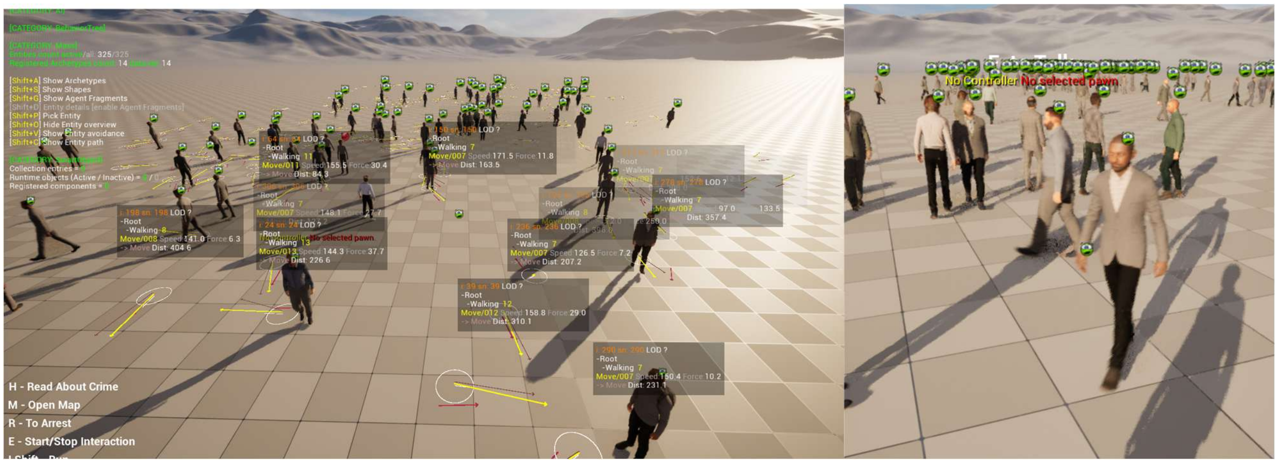


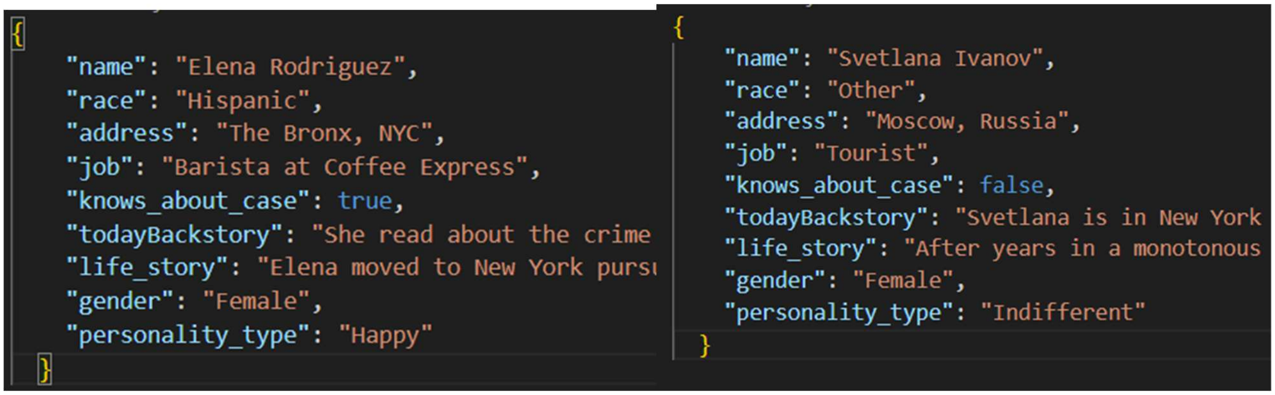
Figure 2 and 3: Shows early-stage crowd performance tests

During the project proposals time, there was no plan to setup AI with Unreal's ECS system as a few NPCs in the level felt more than sufficient at the time, but after setting up a basic crowd system it was observed that a higher number of NPCs always resulted in better range and randomness to the interactions, it also made the enemy harder to find or track and also made the city block look more alive. But Unreal's regular Behaviour tree and AI Controller-based framework had some serious performance bottlenecks as the NPC count scaled up, so a switch was made to Unreal's ECS system called MassEntities. (Figures 2 and 3)

One of the hardest parts of integrating MassEntities into Unreal is the absolute lack of documentation. A significant amount of time was spent reading engine source code and sample projects from Epic and Github to make sense of the implementation. Some things that could have been easily built with the usual behavior trees actually took weeks of effort with the Mass System. It also increased the project's complexity by a considerable amount as anything related to AI had to be computed using Processors with data being set up in Fragment, Traits, and Archetypes. But the upside of all this was the huge scalability boost and performance improvements, so overall in the end it was an excellent choice for the current project as it became easier to scale the NPC count in the level.

Prompting LLMs:

The next step was to write things in regular English and build Unreal C++ systems around it to tune the LLM instances for the game. This proved to be a crucial step to getting the right dialogues and story, the LLM instances when lacked context gave a basic random output that was not useful for the player to solve the case, they also needed to communicate with each other to note down game progress and be constantly "context-aware" throughout the gameplay session. All prompts used in the project are attached as a separate document along with the rest of the project artifacts. The game now has a few prompt instances, a generic prompt that explains the scope of the project, what it is about and etc. A narrative prompt explains the restrictions on the type of stories that the narrative system can generate in the game, similarly, there are Spawn System Prompts, Spatial prompts that explain the direction of the level, and etc.



```
{
    "name": "Elena Rodriguez",
    "race": "Hispanic",
    "address": "The Bronx, NYC",
    "job": "Barista at Coffee Express",
    "knows_about_case": true,
    "todayBackstory": "She read about the crime
    "life_story": "Elena moved to New York pursı
    "gender": "Female",
    "personality_type": "Happy"
}
```

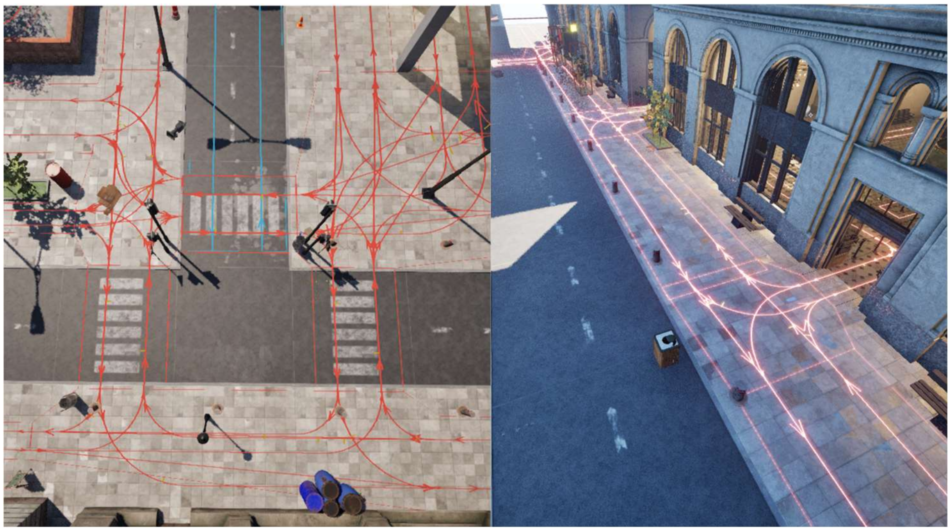
```
{
    "name": "Svetlana Ivanov",
    "race": "Other",
    "address": "Moscow, Russia",
    "job": "Tourist",
    "knows_about_case": false,
    "todayBackstory": "Svetlana is in New York
    "life_story": "After years in a monotonous
    "gender": "Female",
    "personality_type": "Indifferent"
}
```

Figures 4 and 5: LLM generated NPC data JSON

There was also a problem with the quality of dialogues being generated even with expensive LLMs, so since the level was set up in NewYork, a plan was made to integrate parts of dialogues from a popular movie set in New York to teach LLM the way of conversation in that region. This actually solved the problem to a great extent proving once again how important prompting is to set up context. Also for NPC data generation, there were two revelations, one, gpt4o-mini was good enough for this particular task while it fell short for every other LLM query. The images show the quality of output it generated (Figures 4 and 5). Two, LLMs are slow when it comes to generating bigger token size text, for example generating this above data for 100s of NPCs was taking more than 10-15 seconds even for the faster model GPT-4o-mini. So the implementation was changed to more lazy loading NPC data and then attaching them on the go based on race and gender or spawned NPC entities.

Level Design and Environment:

This involved setting up the city block using external 3D assets and setting up zone graphs for the entity systems movement, the Zone Graphs also had rules to set up for turnings, road crossings, one was pedestrian streets, etc. The level was also set up with multiple trigger points to log player and Enemy movements across the city block. Metahuman, City Sample's Crowd, and Mixamo Assets were used to set up the crowd system with the previously built MassAI system.

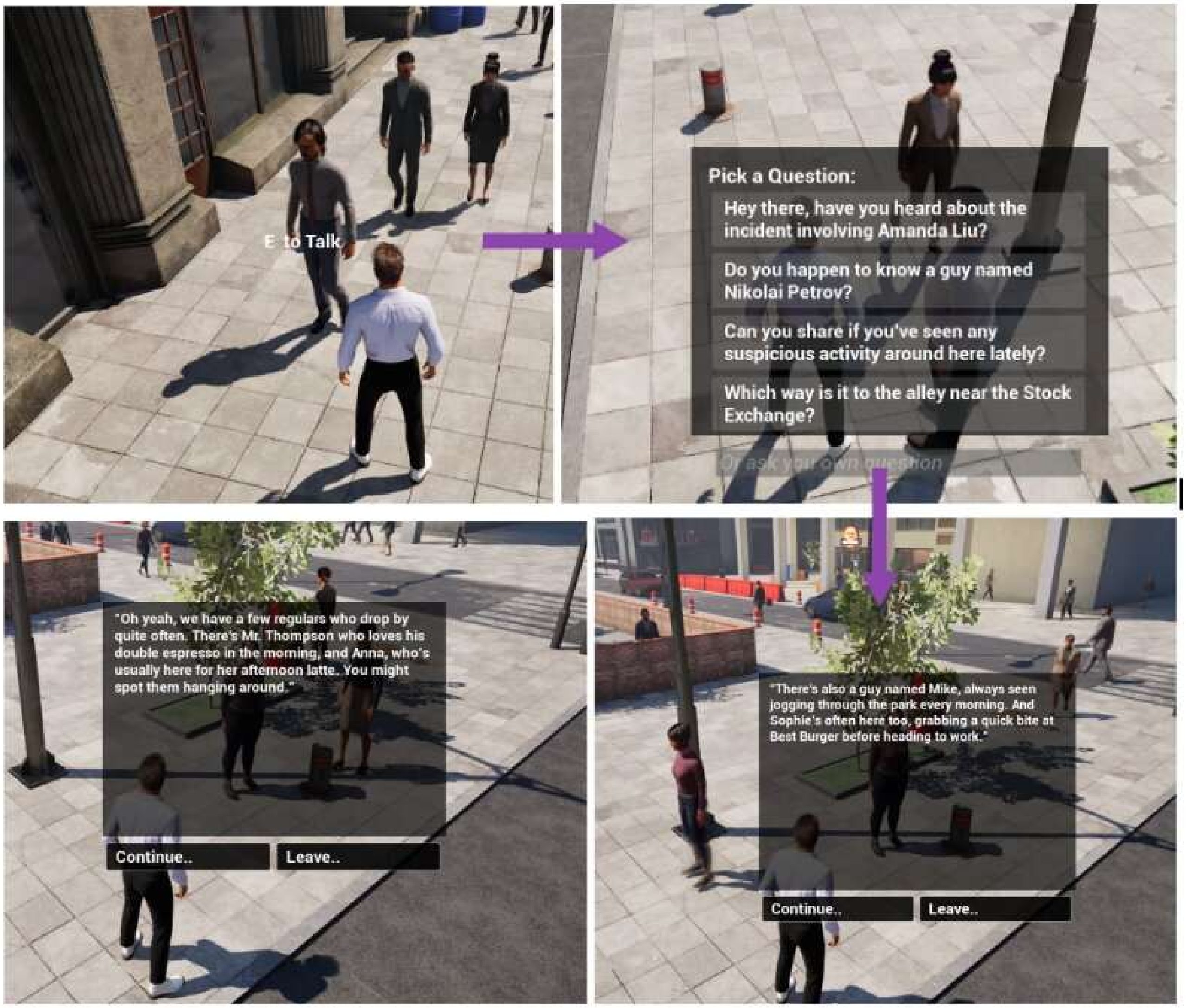


Figures 6 and 7: Zone Graphs for Crowd Movement

The Zonegraphs also had tools to control the flow of NPC “traffic” at junctions and other hotspots, with the foundation set for entity movement there were also initial plans to add a vehicle lane as well, but the process of setting it up increased the complexity of the project and added additional overhead to rendering, so it was later on dropped to focus on crowds alone. (Figures 6 and 7)

Subsystems and Runtime:

The project architecture featured several core subsystems that worked in tandem to deliver a seamless gameplay experience. These included the Narrative Runtime, Agentic Runtime, and Spawn Manager, each of which played an important role in making the game dynamic and context-aware.



Pick a Question:

Hey there, have you heard about the incident involving Amanda Liu?

Do you happen to know a guy named Nikolai Petrov?

Can you share if you've seen any suspicious activity around here lately?

Which way is it to the alley near the Stock Exchange?

Or ask your own question

"Oh yeah, we have a few regulars who drop by quite often. There's Mr. Thompson who loves his double espresso in the morning, and Anna, who's usually here for her afternoon latte. You might spot them hanging around."

"There's also a guy named Mike, always seen jogging through the park every morning. And Sophie's often here too, grabbing a quick bite at Best Burger before heading to work."

Figure 8: Conversation flow with an NPC

The Agentic Runtime subsystem focused on NPC-specific interactions. It enabled NPCs to dynamically decide their actions, such as leaving a conversation or providing misleading clues. This runtime ensured every NPC interaction was unique and relevant to the current game state, enhancing replayability (Figure 8). The Narrative Runtime subsystem handled the overarching story structure, ensuring that the narrative adapted to the player's choices and performance. The Spawn Manager was responsible for optimizing NPC management within the level. It worked with Unreal's ECS system to spawn NPCs.

Testing and Optimisation:

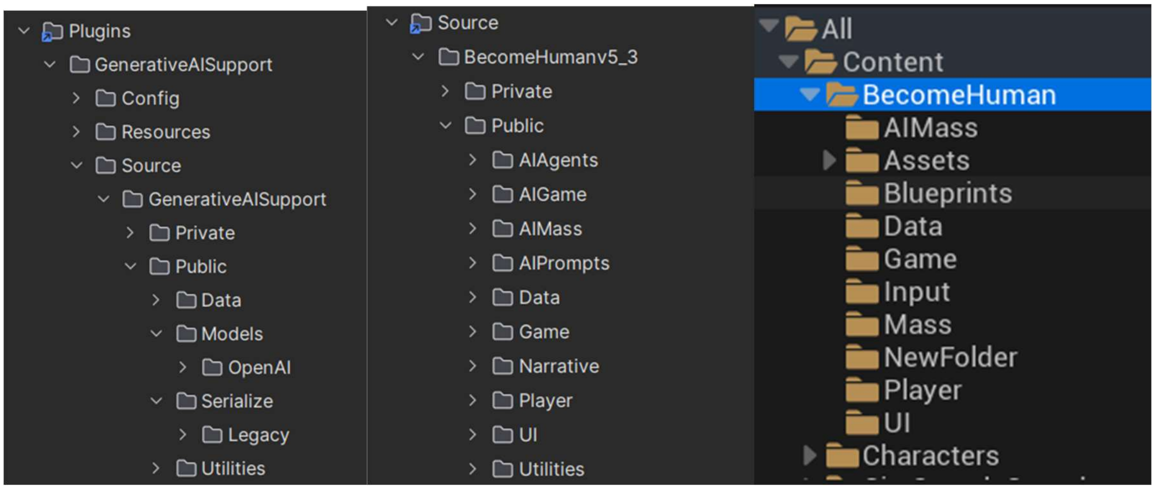
Dialogue generation and narrative progression were tested iteratively, with continuous feedback loops to refine the LLM prompt system. A combination of static and dynamic prompts improved the quality and relevance of NPC dialogues, making them more meaningful for gameplay. Specific tests focused on ensuring NPCs remained context-aware over extended play sessions, or at least they ended the conversation themselves if it went on for a longer time. The crowd system underwent rigorous stress tests to evaluate its scalability. Early crowd performance tests highlighted challenges with frame rate drops when NPC density increased. The Mass System had some LOD actor swap feature for optimisation which mitigated the issue to a good extent.

Tools and Technologies

- **Unreal Engine 5.3:** Picked for its extensive library of game development-related features, full source code availability, easier environment setup, and better ways to set up programming scope by leveraging blueprint and C++ preprocessors and setting up a custom plugin. However, it did add significant complexity to the project because of the sheer volume of features available mainly focused on larger teams.
- **OpenAI API:** GPT-4o for critical narratives and GPT-4o-mini for secondary interactions. As discussed in the earlier section of the document this was picked mainly for its ease of integration, pricing, reliability, and fast response times.
- **Development Tools:** JetBrains Rider for C++ coding, Postman for API testing, and GitHub for version control of the scripts and blueprint excluding any third-party content. The project size was well over 25 GBs before optimizations so it was a struggle to set up proper version control, initially, Perforce was explored too by setting its instance up in the Google Cloud platform but it proved to be increasing the complexity for the current project's timeline.
- **External Assets:** MetaHumans for diverse NPC models, City Sample Crowds for background population, and Mixamo for some additional animations. It was a time-consuming task to bring these different skeletons under a single retarget IK so that some of the animations could be interwoven between the skeletal meshes.
- **Other Tools:** Canva was used to create some of the images in the project, and online formats like Jsonformatter by Curiousconcept and Tokenizer by OpenAI were used to evaluate JSON integrity and cost.

Project Structure:

The project's implementation mainly spreads out in three setups, first is the custom Unreal Plugin developed to handle the LLM API request, it acts like a black box for the rest of the project so that it can be extremely easy to switch to any future model, even from a different organization.



```
v Plugins
  v GeneralAISupport
    > Config
    > Resources
    v Source
      v GenerativeAISupport
        > Private
        v Public
          > Data
          v Models
            > OpenAI
          v Serialize
            > Legacy
          > Utilities
```

```
v Source
  v BecomeHumanv5_3
    > Private
    v Public
      > AIAgents
      > AIGame
      > AIMass
      > AIPrompts
      > Data
      > Game
      > Narrative
      > Player
      > UI
      > Utilities
```

```
All
  Content
    BecomeHuman
      AIMass
    > Assets
      Blueprints
      Data
      Game
      Input
      Mass
      NewFolder
      Player
      UI
  > Characters
```

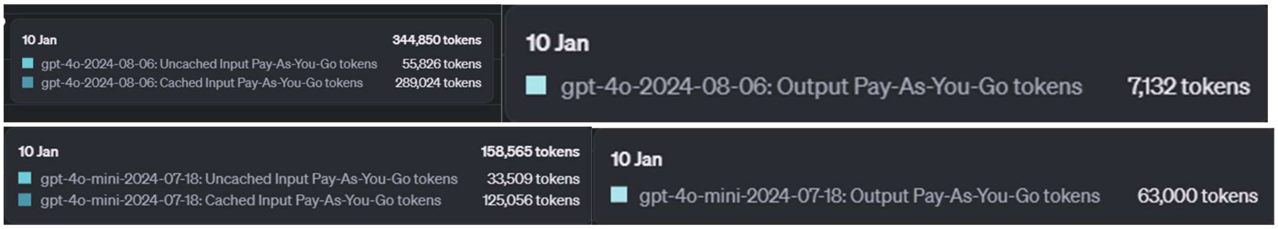
Figures 9, 10, and 11: Project directory hierarchy

The rest of the source code is the main Unreal project, with most of the project's code written in C++ and widget/UI logic written in Blueprint. The blueprints are located in the BecomeHuman folder inside the contents folder in the editor. (Figures 9, 10 and 11)

Outcomes:

The outcomes of the project were evaluated across two primary fronts:

Cost breakdown:



```
10 Jan                                                          344,850 tokens
■ gpt-4o-2024-08-06: Uncached Input Pay-As-You-Go tokens         55,826 tokens
■ gpt-4o-2024-08-06: Cached Input Pay-As-You-Go tokens          289,024 tokens

10 Jan
■ gpt-4o-2024-08-06: Output Pay-As-You-Go tokens                  7,132 tokens

10 Jan                                                          158,565 tokens
■ gpt-4o-mini-2024-07-18: Uncached Input Pay-As-You-Go tokens    33,509 tokens
■ gpt-4o-mini-2024-07-18: Cached Input Pay-As-You-Go tokens     125,056 tokens

10 Jan
■ gpt-4o-mini-2024-07-18: Output Pay-As-You-Go tokens            63,000 tokens
```

Figure 12: Usage reports for a one-hour extensive game session by one player

With **one hour of non-stop heavy gameplay,** the GPT-4o model used about 350k tokens of input tokens out of which ~83% were cached prompts (Openai.com, 2024) and 7k output tokens. In addition to this the same session also consumed 158k GPT4o-mini's input tokens, with again 79% of it being cached prompts. So given the current pricing at the time of the release of the report this would cost approximately **0.57$ for GPT4o** and **0.051$ (with 0.014$ for output and 0.037$ for input) for GPT4o-mini** (Figure 12).

This test session can be considered the worst-case scenario as it was played nonstop for multiple investigations with continuous conversations with the NPC, Narrative Instance, and Spawn instances. So in an ideal case for a regular player, with additional game-side API optimizations, especially for the output tokens and future cheaper LLM launches the cost can quickly go down another 70% to 80%! Although its number is just a prediction, with such a cost it becomes inexpensive to include them in almost every game with some basic OS-wide subscription service like the current PlayStation Plus.

Player Feedback:

Player feedback was gathered through gameplay observation among friends and family. Key highlights included:

- **Quantitative Analysis:**
  - The average playtime was recorded at 12 minutes, a promising start for a prototype.
  - Survey ratings on a 1-5 scale showed an average score of 4.2 for immersion, 3.8 for narrative coherence, and 3.7 for replayability.
  - Metrics on response delays showed an average NPC response time of 1.8 seconds during gameplay, with occasional spikes randomly based on API reliability.
- **Qualitative Analysis:**
  - People appreciated the freedom to explore investigative approaches, leading to diverse outcomes.
  - Areas for improvement were identified, including instances of repetitive NPC dialogues limited NPC-to-NPC interactions, and a lack of crowd-unique animations.
- **Strengths:**
  - Immersive and varied NPC dialogues created a dynamic urban environment.
  - Players appreciated the freedom to approach the investigation in their style, leading to diverse outcomes.
  - The replayable nature of the game encouraged multiple playthroughs to explore alternative story paths.
- **Areas for Improvement:**
  - Instances of repetitive or generic NPC dialogues during extended sessions.
  - Occasional delays in NPC responses during high interaction density, are attributed to LLM processing times.
  - Limited NPC-to-NPC communication, which could enhance world-building and overall immersion.

Ethics: Social, Legal, Data Security:

Legal and Copyright Issues:

1. **Generative AI and Copyright**: The generative AI systems used (OpenAI GPT-4o and GPT-4o-mini) are trained on vast datasets, including copyrighted material. This raises potential legal concerns if the generated dialogue inadvertently reproduces copyrighted text. To mitigate this, the project employs some prompt engineering and content moderation techniques to avoid unintended infringements.
2. **Game Monetization**: The project does not include loot boxes or other monetization features that could be construed as gambling. Any future monetization strategies will adhere to local laws and ethical guidelines, avoiding predatory practices.

3. **Social Stereotypes**: The design actively avoids including harmful stereotypes or offensive content. All NPCs are generated with neutral or contextually appropriate behavior, ensuring a diverse and inclusive experience for players.
4. **Data Privacy and GDPR Compliance**: The game does not store personal data beyond what is necessary for gameplay. Should player data be sent over the network, it would be encrypted and handled in compliance with GDPR and other relevant regulations. Players would also be informed about data usage and given the ability to opt-out.

Ethical Considerations:

1. **Informed Consent**: If the project were to involve external playtesting or player data collection, obtaining explicit consent from participants would be essential. This includes clearly explaining the purpose of the data collection and how it will be used.
2. **Confidentiality**: Any personal or gameplay-related data collected from players during testing must remain confidential. Measures like data anonymization and secure storage mechanisms would ensure that individual privacy is maintained.
3. **Minimizing Harm**: Careful design considerations have been made to avoid offensive or harmful content in NPC dialogues and game scenarios. For example, the game avoids promoting social stereotypes, such as exaggerated or offensive portrayals of NPCs, and ensures that all interactions remain inclusive and respectful.
4. **Conflict of Interest**: The project has no known conflicts of interest, with all technologies and assets used falling under proper licensing agreements or permissions.
5. **Transparency**: Currently it does not disclose to players that the contents are AI-generated. For example, NPC dialogues and responses can be explicitly described as being powered by AI to manage expectations and promote trust, although that would affect the immersion to a great extent.
6. **Right to Withdraw**: In any future commercialization, players should have the right to opt out of any data collection processes without impacting their gameplay experience.

Discussion and conclusions:

The project has successfully demonstrated the potential of integrating Large Language Models (LLMs) into dynamic narrative-driven games. By leveraging Unreal Engine’s Mass AI ECS system and a custom OpenAI plugin, the game achieved realistic NPC interactions and a dynamic storyline. Each gameplay session feels unique even with many ground rules like the player character, and location remaining the same.

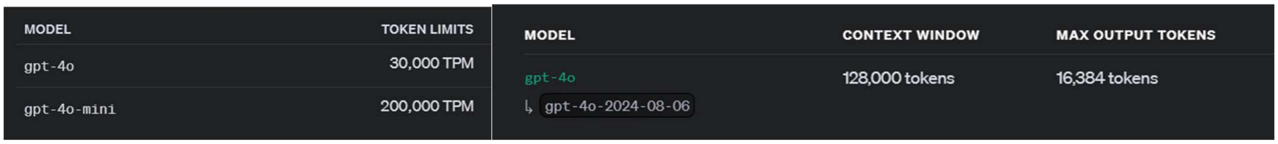
Strengths:

- **Immersive NPC Interactions**: The LLM-driven dialogues provided contextually aware, realistic NPC responses, creating an immersive gameplay experience. The NPCs spoke about the location they were in and changed their dialogues based on their personality type, some knew about the case but were too rude to answer properly, and some others were part of the enemy gang and tried to dodge the question or mislead the player. They also left the conversation on their own will whenever they felt the conversation was going nowhere or the player was asking weird questions. Also having the questions as choices helps in immersing into the conversation for the players especially in the beginning when they were not sure of what they were looking for yet.

- **Dynamic Narration**: The narrative agent showed hints of unpredictable story progression based on player actions, enhancing replayability. The Narrative Instance had the free will to decide when the enemy gang flees hence resulting in players making careful conversation decisions. There were no set rules to play the game in the right way as it was ultimately in the hands of the Narrative Instance to judge the investigation style.
- **Scalability**: The project's setup proved to be extensively scalable, as it leveraged the Unreal Engine's Mass AI ECS system enabled the seamless management of large numbers of NPCs, creating a bustling city environment with next-generation NPCs
- **Cost Feasibility**: The project proves that even the current generation LLMs are relatively cheaper to run a game session like Become Human, especially the inexpensive model like the GPT-4o-mini can be extensively used in the games already with a minimal monthly or yearly subscription like the game pass.

Limitations:

- **NPC to NPC Communication:** One of the biggest improvements that can happen in immersion is to allow NPCs to talk to each other, but this would also come in other kinds of challenges on how many iterations of conversations can happen and the control over API rate limits, etc.



MODEL	TOKEN LIMITS
gpt-4o	30,000 TPM
gpt-4o-mini	200,000 TPM

MODEL	CONTEXT WINDOW	MAX OUTPUT TOKENS
gpt-4o ↳ gpt-4o-2024-08-06	128,000 tokens	16,384 tokens

Figure 13 and 14: Rate Limits set by OpenAI

- **Performance and Rate Limit Constraints**: While optimization efforts improved NPC response times, occasional latency persisted during high NPC density scenarios, and also the project relies on the backend LLM APIs to work all the time, if they go down or if the project exceeds their rate limits then it would be difficult to keep the sessions reliable. The pictures show an example of the OpenAI rate limits for a tier 1 developer account. (Figures 13 and 14)
- **Narrative Coherence**: Even with all the prompts and smart models, there is still some additional coherence that goes mission once in a while from the conversation, or sometimes the LLMs still sound robotic or the replies are mundane. However, this is something that will automatically get better as more and more LLM models hit the market.

Future Work:

1. **Enhanced Optimization**: Maybe the costs can be further cut down when cheaper models launch similar to GPT4o-mini but are also as smart as GPT4o or later. The mass system as

well as the room to further optimize the frame rate and to increase the NPC counts in the level by a large extent.

2. **Deeper NPC Personalities**: Currently the NPCs have a lot of traits like their gender race or occupation, and they look like their gender and race as well, but where it can be improved is the game world impact of their traits, for example, if a NPC is a barista at a local pub if it can be set up in a way they go to the bar every night on workdays and come back late night. Also they get salary credited to their account biweekly which makes them visit cinemas or restaurants on those weekends, etc.
3. **Commercialization Readiness**: To commercialize the project, steps would include additional legal reviews, pricing model adjustments, and scalable infrastructure setup for API management.
4. **Vision API Integration**: Exploring the use of vision APIs for richer environmental storytelling, such as NPCs reacting to specific visual cues in the game world.
5. **Audio:** Audio integration can be a major improvement for the overall immersion.

In conclusion, while the project remains a proof of concept, it opens new possibilities for AI-driven dynamic storytelling in games. With further refinement, it has the potential to revolutionize how narratives are experienced in interactive entertainment.

References:

PlayStation. (n.d.). Discover UNCHARTED. [online] Available at: https://www.playstation.com/en-gb/uncharted/.

Games, R. (n.d.). Red Dead Redemption. [online] Rockstar Games. Available at: https://www.rockstargames.com/reddeadredemption.

Hu, S., Huang, T., Ilhan, F., Tekin, S., Liu, G., Kompella, R. and Liu, L. (2024). A Survey on Large Language Model-Based Game Agents. [online] arXiv.org. doi:https://doi.org/10.48550/arXiv.2404.02039.

Csepregi, L. M. (2023) The Effect of Context-aware LLM-based NPC Conversations on Player Engagement in Role-playing Video Games. Aalborg Universitet.

Gallotta, R., Todd, G., Zammit, M., Earle, S., Liapis, A., Togelius, J. and Yannakakis, G.N. (2024). Large Language Models and Games: A Survey and Roadmap. [online] NASA ADS. doi:https://doi.org/10.48550/arXiv.2402.18659.

Kumaran, V., Rowe, J., Mott, B. and Lester, J. (2023). SceneCraft: Automating Interactive Narrative Scene Generation in Digital Games with Large Language Models. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, [online] 19(1), pp.86–96. doi:https://doi.org/10.1609/aiide.v19i1.27504.

Rao, S., Xu, W., Xu, M., Leandro, J., Lobb, K., DesGarennes, G., Brockett, C. and Dolan, B. (2024). Collaborative Quest Completion with LLM-driven Non-Player Characters in Minecraft. [online] arXiv.org. Doi: https://doi.org/10.48550/arXiv.2407.03460

Mojang (2011). Minecraft official site. [online] Minecraft.net. Available at: https://www.minecraft.net/en-us.

Kalbiyev, A. (2022). Affective dialogue generation for video games. [online] essay.utwente.nl. Available at: https://essay.utwente.nl/89325/.

Bethesda. (n.d.). Fallout 4. [online] Available at: https://fallout.bethesda.net/en/games/fallout-4.

Chen Gao, Q. and Emami, A. (2023). The Turing Quest: Can Transformers Make Good NPCs? [online] pp.93–103. Available at: https://aclanthology.org/2023.acl-srw.17.pdf

Quantic Dream (2018). Detroit: Become Human | Official Site | Quantic Dream. [online] www.quanticdream.com. Available at: https://www.quanticdream.com/en/detroit-become-human.

Warnerbros.com. (2024). WarnerBros.com | The Matrix Resurrections | Movies. [online] Available at: https://www.warnerbros.com/movies/the-matrix-resurrections [Accessed 9 Jan. 2025].

20thcenturystudios. (n.d.). Free Guy. [online] Available at: https://www.20thcenturystudios.com/movies/free-guy

Openai.com. (2024). OpenAI Platform. [online] Available at: https://platform.openai.com/docs/guides/prompt-caching.