ChatGPT

Lagos Stories: AI-First Simulation Project Brief

Author: Jason Ikeokwu

Vision and Concept

Lagos Stories is a text-based **AI-first** simulation game set in a modern-day Lagos city, aiming to push the boundaries of emergent narrative and gameplay. Drawing inspiration from the systemic depth of *Dwarf Fortress* and the life simulation of *The Sims*, it emphasizes realistic urban dynamics over fantasy. The design philosophy is centered on using AI to create a living world where the player has complete freedom and the story unfolds through simulation rather than scripting. Key aspects of the vision include:

- **AI-First Design Philosophy:** Game systems are built with AI at the core. Instead of relying on pre-written plots, Lagos Stories uses AI-driven simulation to generate events, character interactions, and storylines dynamically. This approach treats the AI like an ever-present game master that responds to player actions in intelligent, unscripted ways [1]. The goal is to explore truly improvisational play, where the game can adapt to anything the player tries, rather than funneling them down preset paths.

- **Real-World Lagos Setting (No Fantasy):** The game simulates the **authentic complexity of Lagos**, Nigeria’s bustling metropolis. Neighborhoods, businesses, family units, government bodies, and social ecosystems are modeled after real-world counterparts. There are *no magical or fantasy elements* – the drama emerges from realistic urban life (crime, politics, economy, relationships) rather than mythical beasts or sorcery. This grounded setting differentiates Lagos Stories from typical fantasy simulators and anchors the narrative in relatable reality.

- **Total Player Agency in a Systemic Sandbox:** The player is free to do *anything* their character might plausibly do, and the world’s reaction will be determined by systemic rules and AI actors. There are no fixed win conditions or linear quests – much like in *Dwarf Fortress*, which offers no scripted objectives and lets players freely manage their colony in an open-ended sandbox [2]. Players can choose their own goals (e.g. rise to political power, build a business empire, wreak havoc as a criminal, or just live an ordinary life) and the simulation will support these choices. Every decision, big or small, can have ripple effects across the city, leading to emergent narratives unique to each playthrough [3]. This high level of agency is designed to empower players and encourage creative problem-solving within the game’s systems.

Core Gameplay Structure

The game's architecture is built as a **two-layer simulation model** that balances a persistent world state with focused scenario gameplay. This ensures that the player's actions have lasting consequences, while still providing manageable slices of interaction. The core structure is composed of:

- **World Layer (Persistent City Simulation):** The World Layer represents the entire city of Lagos as a living, breathing environment. At the start of a game, a *persistent world state* is generated based on configurable parameters (via sliders or prompts) such as population size, economic conditions, government type, major organizations, and family demographics. **All NPCs (non-player characters)** in the city – potentially thousands – are generated upfront with unique identities and stored in a database. This world state includes public institutions, businesses, neighborhoods, and social networks. It runs on a slower tick in the background, simulating day-to-day changes, demographics, and high-level events when not in a specific scenario. Crucially, the World Layer provides continuity: it records history and maintains consistency even when the player is not directly involved. (For example, if a new mayor is elected or a company goes bankrupt in the world, that remains true going forward.) By maintaining real persistent state, the game avoids the "ungrounded" feeling that pure on-the-fly generation can produce [4] – events have causes and lasting effects, rather than being forgotten once a scene ends.

- **Instance Layer (Scenario Instances):** The Instance Layer is a **focused, playable scenario** carved out of the larger world. When the player embarks on a storyline or initiates a situation, the game spawns a localized "instance" (e.g. a neighborhood conflict, a family gathering, a crime investigation, etc.) involving a subset of the world – typically 10–30 NPCs relevant to that scenario. The player will control one primary character within the instance, and there will usually be one antagonist or opposing force driving conflict. These instances are where moment-to-moment gameplay happens: the player can move, converse, make decisions, and witness events in a tighter context. Importantly, instances are not standalone levels – they are **slices of the world**. Whatever happens in an instance (say, a character is killed, a business is robbed, a secret is revealed) gets **serialized back into the World Layer's state** once the instance concludes. The world memory is updated so that all NPCs "remember" the outcome and the broader simulation reflects those changes. In this way, instances serve as the stage for interactive drama, while the world provides the persistent memory and context for those dramas.

- **Dynamic Generation & Resolution Phases:** Gameplay alternates between **generation phases** and **resolution phases**. Before an instance starts, there's a generation phase where the game, guided by AI, sets the stage: it pulls relevant NPCs from the world, generates context and objectives for the scenario, and ensures everything is initialized (for example, establishing that in this neighborhood instance, one gang is trying to expand territory while the local shopkeepers hire the player to investigate thefts). After the player's active play in an instance (which might be fast-paced and event-driven), a resolution phase occurs: the outcomes are assessed and merged back into the global world state. This might involve AI summarizing the events and determining knock-on effects (e.g. a successful intervention in a crime ring might lower crime citywide for a time, or a public scandal might erode trust in the mayor). The pace is **fast and real-time within instances** (to keep gameplay exciting), whereas the **world simulation progresses more gradually** outside of instances, allowing deliberate planning and analysis between scenarios.

- **Tick-Based Simulation Loop:** Both layers operate on a **tick-based simulation loop**, meaning time advances in discrete steps (e.g. hourly or daily ticks in the world layer, and perhaps minute-by-minute in an active instance). On each tick, every NPC can evaluate its situation and act according to its AI logic and the game's rules. This approach ensures all entities update in synchrony and enables complex interactions (for example, each in-game hour, a shop might update its inventory, NPCs reconsider their goals, and scheduled events (like work shifts or social gatherings) trigger). The tick rate may differ between layers for practicality, but the principle is consistent. A tick-based loop provides a controlled, measurable flow of time and makes the simulation's progression manageable for developers (and deterministic when needed). It also opens up possibilities for the player to pause and issue commands or fast-forward slower periods, since the game's state only changes on ticks.

In summary, the Core Gameplay Structure marries **global persistence** with **local intensity**. The World Layer guarantees that Lagos feels like a coherent city where everything is connected and memories are long. The Instance Layer ensures the player is never overwhelmed by the entire city at once, but instead engages with it through meaningful episodes. This design lets us have *both* the big picture and the close-up: a persistent world that the player can gradually influence, one focused scenario at a time.

AI Integration and Design Philosophy

AI is not an afterthought in Lagos Stories – it is the engine that drives the game's depth. The design philosophy treats AI as an active collaborator in gameplay, fulfilling multiple roles that traditionally would be handled by scripts or human designers. Key principles of our AI integration include:

- **AI as Interpreter of Player Intent:** The player interacts with Lagos Stories purely through text (natural language commands and dialogue). Behind the scenes, an AI system (likely a large language model or similar) parses the player's input to understand their intent and translate it into game actions or queries. In essence, the AI serves as the **command parser and context-understander**. Whether the player says "convince the landlord to lower my rent" or types a complex plan involving multiple steps, the AI interprets this free-form input and determines what game mechanics or NPC actions that translates to. This goes beyond simple text adventure parsers – the AI can grasp nuance, idioms, or high-level goals and break them down. For example, if a player types, "I want to organize a block party to improve community morale," the AI could orchestrate the necessary steps: gathering NPCs, allocating resources, initiating a social event in the simulation. By using AI in this way, we allow a much wider range of player expression than a traditional menu of actions, making the experience feel closer to interacting with a human game master.

- **AI-Driven Content Generation:** Lagos Stories employs AI generation to create rich, **nuanced world and instance content** on the fly. This includes dialogue lines for NPCs, descriptions of locations or events, backstories for characters, and even procedural quest-lines – all tailored to the current context. The key is that while the world layer provides facts and constraints (e.g. NPC A is a taxi driver with a greedy trait, NPC B is her brother who has a feud with a gang), the AI can use these to **generate flavorful details** that bring scenes to life. For instance, two NPCs might start arguing about a past incident dynamically, or an AI might inject an unforeseen complication in an instance (like a sudden power outage or a surprise visit from an in-law) that wasn't explicitly scripted but fits the scenario. The AI content generation is kept **bounded by the simulation state** – it cannot contradict known facts stored in the world model (ensuring consistency), but it has freedom to elaborate within those bounds. This structured use of AI (sometimes called “structured generation”) ensures creativity without incoherence [5] .

- **Simulated Character Agency via AI:** Each NPC in the game is essentially an **AI agent** with its own goals and decision-making capability. We leverage AI techniques to give NPCs a degree of *autonomy* and unpredictability in how they pursue their objectives. Rather than using only deterministic behavior trees or simple state machines, characters can have AI-driven behavior that reacts adaptively. For example, an ambitious NPC politician might, entirely on their own initiative, begin forming a coalition to oust a corrupt official if the opportunity arises, or a teenager NPC might start an underground music club as a response to the lack of creative outlets. These are not hard-coded “quests” but emergent initiatives flowing from the characters’ motivations. The AI components look at an NPC’s profile (traits, relationships, needs) and the state of the world, and **generate intents or actions** that the NPC might take. This results in a world that feels alive and dynamic independent of the player – factions maneuver, gossip spreads, crime rings form or collapse – all driven by AI simulating each character’s mind. Notably, this includes antagonists: if the player is trying to reduce corruption, an AI-driven antagonist might react by covering their tracks or targeting the player’s allies, even if the player never directly engages them. The use of AI in this way means the game can produce scenarios far beyond what we manually imagine, increasing replayability and surprise. *(For instance, testers of an AI-driven game reported NPCs spontaneously organizing a hide-and-seek game when bored [6] – a perfect illustration of unscripted character agency.)*

- **AI-Augmented Game Mastering:** In Lagos Stories, the AI also plays the role of an **orchestrator** or *dungeon master* that guides the narrative pacing. It can introduce new story threads based on player input or emergent world events, effectively suggesting “what happens next.” If the player’s actions create a power vacuum in a district (say they took down a local gang), the AI may decide an opportunistic character tries to fill that void – generating a new adversary or challenge for the player. Conversely, if the player seems to be struggling or aimless, the AI can surface hooks (like having an NPC approach the player with a dilemma or an opportunity) to gently guide engagement. This adaptive storytelling ensures there is always something interesting going on that ties logically into the world’s state. We emphasize that **AI is used thoughtfully** – it’s not just generating random events, but acting as an intelligent director that maintains narrative momentum and cohesion. Industry pioneers have posed similar questions about AI as game master, e.g., asking whether generative AI could interpret player actions and make rulings like a human Dungeon Master [1] . Lagos Stories is built on the belief that the answer is “yes” – with AI, we can let players truly forge their own path while the game remains responsive and coherent.

- **Examples of AI-Driven Gameplay:** To concretely illustrate, here are a few scenarios highlighting AI integration:

- *Dynamic Crime Network:* Suppose the player character focuses on their small business and ignores rising crime. The AI, noticing a lack of interference, might have an NPC gang expand into a organized drug network in the city. New events spawn: a string of drug-related incidents, NPCs whisper about where the drugs are coming from, and an undercover cop NPC approaches the player for help. None of this is pre-authored – it emerges because the conditions allowed it and the AI wove a narrative thread around it.
- *Social Reputation & Rumors:* If the player does something notable (heroic or notorious), AI-driven NPCs could start spreading rumors. The tone and content of the rumor (admiring, fearful,

exaggerated, etc.) can be generated based on the NPC's perspective and relationship to the player. These rumors then affect how other characters treat the player (simulation ties in), and can even spark new mini-plots (e.g. a rumor about the player hoarding wealth might attract a thief).

- *Adaptive Challenge:* The game monitors the player's progress. If the AI notices the player has become very influential in local politics, it might *escalate* the challenge by generating an antagonistic event – say a coalition of rival politicians scheme to undermine the player (complete with AI-written smear campaigns in the news or a surprise audit of the player's business). On the flip side, if the player is struggling, the AI might engineer a helpful event, like an old friend (NPC) offering resources or advice.
- *Player-Led Ideas:* Perhaps most importantly, if the **player themselves proposes something unconventional** ("What if I start a city-wide charity event?" or "I want to rally a neighborhood watch to fight crime"), the AI systems will attempt to accommodate it. The AI can generate the sequence of tasks or responses needed – maybe the player needs permits, volunteers, and publicity for the charity, and each of those spawns sub-interactions. By interpreting intent and leveraging content generation, the game *encourages creative strategies* instead of replying "You can't do that." In essence, **if you can describe it, the game will try to do it**. This level of freedom is a core promise of the AI-first design.

In summary, AI in Lagos Stories is the secret sauce that turns a static simulation into a responsive narrative experience. It is carefully structured – the AI works within the rules of the world model – but it provides a flexibility and intelligence that allow the game to feel like a true storytelling partner. We're not using AI just for cheap content generation; we're using it to unlock gameplay that was previously impossible: a city that feels truly alive and a game that can **improvise** along with the player.

Simulation & World Modeling Systems

To support the rich gameplay and AI-driven events, Lagos Stories requires robust simulation systems modeling both individuals and the larger society. Every character and entity in the game has a detailed representation in the world model, ensuring their behavior and interactions can be simulated with depth and consistency. Key components of our world modeling include:

- **NPC Profiles (Traits and Alignment):** Each NPC is more than just a name – they come with a profile defining their personality, alignment, and core traits. This could include D&D-like alignment (lawful, neutral, chaotic; good vs evil) or a more nuanced moral compass, as well as traits such as *ambitious, greedy, compassionate, introverted, traditional, volatile,* etc. These traits influence how the NPC will behave and respond to situations. For example, an *aggressive* NPC might be quicker to resort to violence when threatened, while a *compassionate* NPC could be more forgiving or helpful. Alignments provide a high-level guide to an NPC's values (e.g. a law-abiding citizen vs. a corrupt opportunist). These attributes feed into the AI's decision-making for that character. In simulation terms, traits act as parameters that skew an NPC's choices under the game's general rules. This ensures that not all NPCs act the same – their individual "personalities" lead to diverse actions and reactions, making the world feel populated by distinct people.

- **Relationships and Social Networks:** Relationships between characters are explicitly tracked. NPCs can have family ties, friendships, romances, rivalries, mentorships, and more. There are also group affiliations: membership in organizations, companies, gangs, political parties, neighborhoods, religious groups, etc. Each relationship can have a strength or sentiment (love, hate, loyalty, distrust) associated with it. The simulation leverages these relationships to propagate effects: for instance, if an NPC is harmed, their close friends and family will react (perhaps seeking justice or retribution). If someone gains power, their allies benefit and their enemies suffer. Relationships also allow for emergent coalition behavior – e.g., several NPCs who all dislike a certain character might team up against them. We will model *value flows* through these networks too (information, money, favors, influence can spread along social connections). This is akin to how *The Sims* models friendships and family, but here it's extended to a city-wide scale with factions. Every NPC will thus have a "social context" that shapes their behavior: who they trust, who they obey, who they compete with.

- **Skills, Resources, and Status:** NPCs have skill sets (e.g. a doctor has medical skill, a hacker has tech skill, an athlete has physical skill) which determine what actions they can effectively perform. They also possess resources (money, property, access to assets) and possibly special items or information. Status attributes like health, stress, or reputation might be tracked as well. These factors feed into simulation rules: e.g., an NPC with high wealth can bribe others; one with hacking skill can exploit certain opportunities; an NPC in poor health might become desperate or seek medical help. Resources also constrain actions (no money means an NPC might turn to crime or ask for help). By tracking these, the simulation ensures logical consistency – a random cab driver won't start a high-tech company without resources or skills, unless they acquire them somehow through gameplay.

- **Loyalties and Faction Alignment:** Beyond personal traits, NPCs can hold loyalties to larger entities like factions or ideologies. One might be a loyal member of the police force, another secretly loyal to a criminal cartel, another to their ethnic community or hometown. These loyalties affect decision-making: an NPC might sacrifice personal gain to help their faction, or betray a friend for what they see as a higher cause. Factions and organizations themselves (which can include city government, corporations, gangs, NGOs, community groups, etc.) have their **own attributes and value systems**. For example, the city government entity might have an "ideological" value system prioritizing law & order and public approval, whereas a crime syndicate's value system is economic profit and survival. These values determine how these entities (through their NPC members) react to events. The interplay of individual desires and organizational loyalties creates complex motivations – e.g., a police officer NPC might personally sympathize with a protest cause (personal trait), but their loyalty to the police force and orders from superiors (faction loyalty) pull them the other way. The game's simulation will model these tensions, often forcing characters to make choices, which in turn generates drama.

- **City-Level Systems (Economy, Politics, etc.):** On a macro scale, Lagos itself is simulated through various systems. The economy system tracks markets, employment, cost of living, and trade – e.g., if unrest leads to businesses closing, unemployment rises and crime might increase, etc. The political system tracks things like the current leadership, public order, legal policies, and civic issues. For example, a city-wide "corruption" level might be simulated and influenced by in-game actions; high corruption could lead to lower public trust, protests, or external government intervention. Other potential systems include infrastructure (power, transportation), education, public health – each of which can produce scenario hooks (a power outage, a traffic jam affecting an instance, a disease outbreak). These systemic values provide a backdrop for instances and can evolve over time based on what the player and NPCs do. We ensure these are *distinct value systems* – e.g., economic changes affect money and goods, whereas ideological changes affect public opinion or NPC behavior – but they interconnect (a bad economy might fuel political unrest, etc.). Modeling these helps create a sense that the world doesn't revolve solely around the player; it has its own momentum.

- **Historical Memory and Persistent World State:** The game keeps a **history log** of significant events and changes in the world. This means if an NPC becomes the mayor in year 2025 and then is assassinated in 2026 during gameplay, that fact remains part of the world's history, and NPCs will remember it in conversation or in their current world view. Every in-instance action that is notable gets recorded: crimes, public acts, relationship changes, even rumors. This historical memory is used to inform AI decisions and narratives. For example, if years later the player encounters a new character, that character might mention "I remember the day our former mayor was shot; it changed everything." Persisting these details enables long-form storytelling. The world state is effectively a **database of facts** about all characters and events. NPC AI queries this to decide what they know or how they should feel about something. This supports emergent storytelling across multiple play sessions or timelines – if you play over decades of in-game time, you can witness a true saga unfold with consistency. In procedural games like *Dwarf Fortress*, players have lauded how rich histories emerge from simulation [3], and Lagos Stories aims for a similar effect in a contemporary setting. Small actions can snowball into major historical events in the simulation.

- **Playable Character Perspective:** While the world runs all NPCs, the player can in some cases **shift perspective** or choose new characters to play as time goes on. You might start as one character (say a young journalist) and follow their life to old age. If that character dies or you want a change, you could take control of another character in the world (perhaps a relative or someone influential affected by the first character's actions). This ability to **jump between characters or generations** means the game isn't limited to one protagonist story – you can explore the world from many angles. It's somewhat akin to how *Crusader Kings* lets you continue the game as your heir when your ruler dies, or how *The Sims* allows switching the controlled household. This design supports exploration of the full world simulation: you can follow a single character's arc or experience how different lives intersect in Lagos. The simulation supports this by keeping the world state continuous; when you switch characters, the world remains as it was, with all previous characters and changes intact. Persistent storytelling is thus not just across time, but across points of view.

In summary, the simulation systems of Lagos Stories strive to capture the **interconnectedness of an urban society**. Every NPC has depth and ties to others; every faction has goals and influence; every action reverberates. We are effectively building a *text-based simulation of Lagos city life*, where narrative emerges from the interactions of all these components. The level of detail (from personal relationships to city economics) provides a sandbox for the AI and the player to generate endless stories. As the City 20 project similarly noted, giving every citizen their own motivations and relationships results in no two stories being the same [7] – that is exactly the outcome we seek: a game that produces unique narratives grounded in a realistic simulation of human society.

AI vs System Balance

A critical design challenge in Lagos Stories is balancing the power and freedom of AI-driven simulation with the need for coherent, player-centric gameplay. With AI deeply involved, we must set boundaries and decide how much agency to cede to autonomous systems. Here we outline the key considerations and open questions regarding this balance:

- **NPC Autonomy vs. Player Agency:** How much freedom should AI-controlled NPCs have to initiate major actions on their own? On one hand, high autonomy means a more lifelike world (characters will plot, change the world, and surprise the player). On the other hand, if NPCs do too much *without* the player's involvement, the player could feel sidelined in their own story. There have been instances in AI-driven prototypes where NPCs "left the player with little to do" by solving problems independently [8]. We want to avoid a scenario where, say, the player plans to resolve a conflict but an AI NPC already resolved it off-screen. Therefore, we need to balance it such that NPCs advance their agendas, but significant world-altering events give the player a chance to intervene or at least witness them. The game might, for example, require the player's presence for the climax of major story events, or generate new challenges in response to NPC actions to keep the player engaged as the hero (or anti-hero).

- **Antagonists and Off-Screen Activity:** Should antagonists (or any NPC) be active "behind the scenes" outside of instances? For realism, we'd like antagonists to scheme continuously – e.g., a crime lord might be expanding his empire even while the player is focused elsewhere. This creates a feeling of a living world. However, if the game advances too much off-screen, the player could be blindsided or feel like they have no control. One design option is to have **asynchronous simulation** for off-screen NPC goals: they progress, but maybe slower or with periodic checks that give the player opportunities to discover or foil plots. Another option is to only simulate complex antagonist moves during instance gameplay or as scheduled events that the player can opt into. We need to decide how "active" the world is when the player isn't looking. A middle ground might be: everyday life goes on for NPCs continuously, but high-impact plots either wait for the player or are telegraphed via clues (so the player can choose to get involved). Essentially, do we allow stories to advance without the protagonist, or pause them until the protagonist is present? This is a balance between world simulation integrity and dramatic timing in gameplay.

- **Continuous Simulation vs. Event-Driven AI:** Related to the above, we must determine how frequently the AI logic should run, especially given computational costs. One approach: **per-tick AI**, where every tick, every NPC potentially thinks and acts (this is truer to life but extremely heavy if using complex AI models for each NPC continuously). Another approach: **event-driven AI**, where the AI comes into play at key decision points or triggers (e.g., when the player directly interacts, or when a global event threshold is reached). Perhaps a hybrid works: use lightweight rule-based simulation for routine NPC behavior each tick, and invoke the heavy AI routines only for salient moments (like an NPC making a big life decision or responding to a player's unusual request). We also consider using **simplified simulation models** for background activity, reserving full AI reasoning for foreground interactions. The trade-off is between fidelity and performance/coherence. Continuous AI might yield very emergent outcomes but could be unpredictable and costly. Event-based AI gives more control and stability but could miss out on subtle emergent happenings. We will likely iterate to find the right balance, possibly scaling AI detail with proximity to the player (e.g., NPCs near the player or in relevant story arcs use detailed AI, others further away use coarse simulation).

- **World Coherence and Narrative Control:** One fear with AI-driven content is that it can introduce chaos or contradictions if not managed. We need to enforce a level of **coherence** in the world's narrative. This means establishing ground truth in the world state that AI must respect (no sudden alien invasions or out-of-character behavior unless it's been logically set up in the narrative). We may implement an *adjudication layer* – essentially rules that check or veto AI-generated events if they break consistency or design constraints. For instance, if an AI tried to spawn a city-wide riot out of nowhere, but there was no preceding cause or player action related to unrest, the system might reject or modify that suggestion. The design question is how much *freedom* to give the AI to introduce dramatic twists versus how much to tightly control story arcs. We want the world to feel organic, but also to produce satisfying story structure. Too much control, and we lose the AI's benefit of surprise; too little, and the narrative could become incoherent or overwhelming. We will explore techniques like **AI alignment** (making sure the AI's goals are aligned with overall game narrative goals) and use of *director* algorithms (similar to Left 4 Dead's AI Director, but more general) that ensure the pacing of events remains engaging. Essentially, the AI should be creative, but still *play by the rules of our world*.

- **Player Freedom vs. Guidance:** Another axis of balance is how to handle the **player doing the unexpected**. Since players can attempt anything in text, our AI and simulation should be able to handle left-field ideas. We want to encourage this freedom – it's a major selling point. However, we also must prevent the player from inadvertently breaking the game's logic or getting lost. We won't ever tell the player "you can't do that," but the game might sometimes need to *gently steer* the experience. For example, if a player action would completely derail the simulation or trivialize all challenges ("I burn down the entire city in one go"), the game might channel that impulse into a smaller scale outcome or present realistic barriers (it's not so easy to do that overnight). Likewise, if a player is stuck, the AI might suggest a course of action in-character ("Your friend suggests maybe talking to the district chief about this issue."). The open question is: **where to draw the line between freedom and guidance?** We want to avoid heavy-handed "railroading," but we also want the experience to have some structure that players can follow. Playtesting will be crucial here – finding if players feel overwhelmed by choice and need more direction, or if they feel too constrained. Achieving a satisfying balance will likely involve UI/UX solutions as well (surfacing options without limiting creativity).

- **Ensuring Fun Emergence (AI Alignment with Game Design):** As an aside, there's the issue of AI being *too realistic*. A completely rational simulation might result in NPCs that avoid risk, or conflicts that resolve too sensibly – which can actually reduce the fun (as observed in one case where AI NPCs became so helpful to each other that players had nothing to solve [8] [9]). We'll likely have to tune the AI's objectives to ensure a steady supply of conflict, mystery, and challenge. In other words, the simulation might need a bit of *drama bias*. Real humans might avoid a dangerous confrontation, but our AI characters, to keep things interesting, might take slightly more extreme actions than an average person – just enough to generate engaging scenarios. This is part of balancing simulation authenticity with entertainment. It ties back to the concept of an *AI Director*: sometimes, to maintain a coherent game, we might override or influence an NPC's pure simulated inclination for the sake of a better story. The design will involve deciding those boundaries and documenting rules for the AI (for example: "if an NPC is an antagonist, allow them more bold/antagonistic actions than their personality might strictly dictate, to drive gameplay").

In summary, **AI vs System Balance** is about finding the sweet spot where the AI is neither running amok nor muzzled to the point of dullness. We want the world to have a life of its own, yet the player should feel *essential* and see the impact of their involvement. We also aim for emergent surprises, yet within a framework that feels coherent and intended. These design questions don't have one right answer up front – they will be iteratively answered through prototyping and playtesting. Ultimately, our philosophy is to err on the side of giving the player and AI freedom to experiment, while putting guardrails in place to catch truly disruptive outcomes. The result should be a game that is *alive and unpredictable* but still delivers a satisfying narrative arc that players can follow.

Open Questions and Design Decisions

While we have a strong high-level concept, there are several open design questions to address as development proceeds. These are areas that require further research, prototyping, and possibly input from domain experts (AI researchers, game designers) to refine. Our current key open questions include:

- **Defining Core World Values:** *What fundamental metrics or "values" should the world simulation track?* In other words, what are the primary quantitative or qualitative values that drive the world's state? Possibilities include economic wealth, social stability, public happiness, corruption level, security/crime level, etc. We need to decide which of these are most critical to model for Lagos Stories. The choice will influence both gameplay and AI behavior (e.g., if "public order" is a value, then events can raise/lower it and NPCs might act to restore order if it's low). A related design decision is how these values are represented – as numerical scores, categorical states, or something else. For instance, do we have a single "city prosperity" index, or do we separately track employment rate, average income, etc.? Striking the right balance in complexity is key: enough to capture meaningful changes, but not so many that the simulation becomes unmanageable or opaque to players.

- **Multi-Tier Value Systems (Individual vs Collective):** *How do we structure different value systems across levels of society?* An individual NPC has personal values (e.g., their own happiness, moral code), an organization might have collective values (profit for a company, influence for a political party), and the civic level has societal values (e.g., overall crime rate, cultural norms). We need a framework for how these interact. For example, if a company values profit, it might encourage its member NPCs to act greedily, which might conflict with those NPCs' personal ethics if they value honesty. If the city as a whole values safety, that could conflict with a gang faction's goals, creating tension. The design question is: do we explicitly implement different "currencies" or variables for each (like Personal Satisfaction, Faction Power, Public Welfare), and if so, how do actions convert between them (e.g., a criminal act might increase one character's wealth value while decreasing the public safety value)? We also consider whether any unified value (such as a generic "influence" or "utility") can be used, or if a vector of values is needed to capture trade-offs. This decision affects AI decision-making: an AI character might have to weigh personal gain vs. loyalty to a group vs. legal risk – we need to give the AI a way to evaluate those. Designing this system of values and trade-offs is an open challenge, potentially informed by fields like sociology, economics, or game theory.

- **AI's Role in the Tech Stack:** *Where exactly does AI fit into the game's architecture and loop?* We have identified several roles (content generation, intent interpretation, character behavior, narrative directing, outcome adjudication). The open question is how to structure the game engine to integrate these smoothly. Options include: having a centralized AI module that takes the game state and returns narrative outcomes (a black box approach), versus embedding AI at multiple points (e.g., NPCs each have an AI process for decisions, a separate AI for dialogue generation, another for world events). We must decide which tasks are best handled by AI and which by deterministic simulation. For example, should combat or physical outcomes be resolved by physics/logic (to ensure fairness), with AI only describing flavor? Or should AI decide if an attack succeeds based on context? Similarly, do we use AI to generate the world initial state, or do we algorithmically generate it and only use AI for storytelling on top of it? There's also the question of **online vs offline AI**: some content might be pre-generated (offline) for efficiency, whereas interactive dialog must be online (real-time). Determining a clean architecture – perhaps a pipeline where the AI is invoked at certain triggers – is an ongoing design task. Essentially: which parts of the game rely on learned AI models, and which parts on traditional code and data? Making these boundaries clear will help us ensure stability and make future extension easier.

- **Extensibility and Modularity:** *How do we design the system to allow future extensibility while maintaining structure?* We want Lagos Stories to be a framework that can grow (adding new event types, new AI behaviors, new content) without breaking. To do this, we need to enforce some structure or standards in how content is generated and stored. One idea is to use **templating or schema-based generation** for certain content – for example, have templates for common event types (a robbery, a political scandal, a wedding, etc.) that AI can fill in with specifics. That provides consistency in how those events are represented in the game state, making it easier to add more or debug. We also might expose modding hooks or data-driven definitions for things like NPC types, neighborhood properties, or mission templates, so designers (or even players) can extend the game. The challenge is doing this without constraining the AI's creative potential too much. Another aspect is how to manage updates: if the AI model improves or we change a system, how do we ensure it doesn't invalidate the world state or break prior content? We need a strategy for versioning or translating old state into new rules if the simulation rules evolve. In short, we're asking: how can we architect the game such that it's **modular** (each system component can be worked on independently) and **extensible** (new features can plug in), all while keeping a coherent overall design? This often means clearly defining interfaces between systems (e.g., AI can propose an event, but that goes through a "validation module"; or an NPC AI can only act via defined action APIs). Clarifying these boundaries is an ongoing design consideration that will be guided by prototyping experience.

- **Player Experience and Learning Curve:** (Though not explicitly listed, it's worth noting as a design question) How do we teach players to play this kind of game and not overwhelm them? Games with high freedom can cause "analysis paralysis" or confusion about goals. We likely need to design an adaptive tutorial or early-game guidance to show what's possible without hand-holding. The open question is: what is the best way to introduce the AI-driven mechanics to players so they understand they really can try anything, and how do we communicate the state of this complex world in a UI? This overlaps game design and UX – perhaps providing tools like an in-game journal or newspaper that summarizes key world events, or an AI "advisor" character that gives hints. As we build the game, we'll keep asking how to surface the right information to players at the right time, and how to let them set their own goals without feeling lost.

These open questions highlight that Lagos Stories is breaking new ground, sitting at the intersection of simulation, storytelling, and AI. They will be addressed through continuous design iteration and likely collaboration with experts or the community. By keeping these questions in mind, we ensure we remain flexible in our approach and focus our R&D efforts where it matters. Not everything can be solved at once; part of the next steps will be identifying which of these unknowns to tackle first, and which can be refined as the project evolves.

Next Steps

To move forward with Lagos Stories, we propose a phased approach focusing on prototyping core systems, evaluating technology options, and gradually building up the full experience. Below are the recommended next steps, along with tools and future areas of exploration:

- **Prototype Phase – Core Loop Proof of Concept:** Begin by prototyping the fundamental game loop on a smaller scale. For example, implement a mini-simulation of a single neighborhood in Lagos with, say, 20 NPCs. Include a basic world layer (the neighborhood with a few locations and simple city stats) and an instance layer (perhaps a scenario like a neighborhood conflict or mystery). Use this prototype to test the **AI integration** end-to-end: the player inputs text, an AI interprets it, NPCs react on a tick, and outcomes are saved. Focus on one or two key mechanics (e.g., conversation and one type of conflict) to keep it manageable. This prototype will help validate the AI-first approach and highlight any glaring issues in the AI vs system balance (for instance, do NPCs behave coherently? Does the AI produce sensible output consistently?). It will also let us gauge performance and cost in a microcosm before scaling up. Success in the prototype phase is defined by achieving a playable "slice" of the game where an emergent story can unfold over, say, 15-30 minutes of gameplay.

- **Technology and Tooling Decisions:** Based on our needs (text-heavy interface, simulation backend, AI model integration), we evaluate engines and frameworks. **Godot 4** is a strong candidate for the game engine: it's open-source (no licensing issues), flexible with 2D/UI which is suitable for text-based or simple graphical representations, and has support for integrating external libraries (for AI inference or database). Godot's GDScript or C# could be used for simulation logic, and we can interface with Python or C++ for AI components if needed. Another reason to favor Godot is its support for custom tools – we could build in-editor tools for designing NPC templates or event templates, aiding development. On the AI side, we must choose our language model backend. Given the desire for possibly local/offline execution (for lower cost and data control), we should experiment with open-source LLMs. **Mistral** (an emerging open model known for efficiency) or derivatives of **Llama 2** are promising options. We can use **llama.cpp** or similar libraries to run these models on local hardware (or player's machines) with optimized performance. This avoids recurring API costs and latency of cloud calls – a lesson learned from other AI game projects where cloud inference was prohibitively expensive [10]. Initially, we might prototype with a smaller model (for speed) and gradually move to larger ones as needed, always keeping an eye on optimization. Part of this step is also setting up a pipeline for AI prompts and responses within the game loop (e.g., designing prompt templates for the AI that include relevant game state, and functions to parse the AI output reliably). Additionally, establishing a database for world state (could be as simple as JSON or an SQLite database in early stages) will be important for persistence.

- **AI Model Considerations:** We should start with relatively small models (7B-13B parameter range) that can run reasonably fast on available hardware, then scale up as necessary. If targeting local execution, quantization techniques (like 4-bit quantization for LLMs) can help fit models in memory. If we find that larger models are needed for quality, we might look into a hybrid approach: use a local model for frequent, simple tasks and occasionally call a cloud-hosted big model for complex narrative generation (though we'd prefer to avoid this ongoing cost if possible). Fine-tuning or training custom models on game-specific data (such as Lagos lore or dialogue style) is a future possibility to improve coherence and reduce undesired outputs. We should also design the system to be model-agnostic where possible, so we can swap out or upgrade the AI component without rewriting the whole game logic.

- **Future Feature Development:** Once the core loop is proven, we can expand the scope. Future development areas and research priorities include:

- **Psychology & Personality Modeling:** Enhance the NPC profiles with more complex psychological models. This could involve attributes like motivations, fears, long-term goals and even simulation of emotions or mood that fluctuate with events. For instance, modeling the *Maslow's hierarchy of needs* for each character could influence their behavior (if basic needs aren't met, that dominates their actions). We might collaborate with AI psychologists or use existing models (like the *Five-Factor Model* of personality) to enrich NPC behavior patterns.
- **Economic and Systemic Simulation:** Build out a more detailed economic simulation of Lagos – include currency, prices, transactions, and perhaps a simple market system. This will allow stories around business, poverty, wealth, etc., to play out more realistically. Similarly, simulate other systems like law enforcement (chance of police response to crimes), environment (traffic, utility outages), etc. Each of these systems adds layers of realism and new event types (e.g., a simulated stock market could lead to insider trading story; a health system could lead to hospital dramas, etc.). We must ensure these systems integrate into the AI's understanding (so it knows the context of a stock market crash, for example).
- **Advanced Natural Language Understanding:** While the AI will handle a lot of understanding, we may incorporate NLU techniques or custom parsing to handle certain commands more reliably. For example, a predefined set of verbs (like "go to [place]" or "give [object] to [NPC]") could be recognized with high precision to reduce misunderstandings. Additionally, developing a more structured **ontology** of game concepts (locations, people, items, actions) and making sure the AI is aware of it via prompt engineering or fine-tuning will improve consistency. Research into semantic parsing or dialog management from interactive fiction could be useful here. Essentially, we want the game to robustly handle both free-form input and more standard commands. We might also consider multi-turn conversation flows where the AI can ask the player for clarification if needed (like a NPC saying "Do you mean you want to do X or Y?").
- **Content Templates & Tooling:** To maintain structure in a largely AI-generated game, we will benefit from creating templates or parameterized event scripts. These are like skeletons of events that AI can flesh out. For example, a "betrayal" event template might specify roles (betrayer, victim, context) and consequences, and the AI would fill in the narrative details given specific characters and reasons. By having a library of such templates (covering common dramatic scenarios like betrayal, alliance, discovery of secret, chase, etc.), we ensure the game hits satisfying dramatic beats even as details vary. We will invest in tools to author and test these templates, and possibly a system for the AI to select which template suits a situation (this could be another AI classification task). Template-based generation can greatly help in keeping the AI-generated content coherent and also easier to debug (since we have a known structure). Additionally, developer tooling for visualizing the simulation (seeing NPC relationships graph, or timeline of events) will be important as the project grows – these help us understand and fine-tune the complex emergent behaviors.

- **User Interface & Presentation:** Though the game is text-based, we should explore ways to present information elegantly. Perhaps a mix of text and simple graphical UI (maps of the city with markers, character portraits or relationship diagrams, etc.) to help players digest the state. We might incorporate an in-game smartphone interface for the player character (as a diegetic UI) where they can see news feeds or messages that summarize world events – essentially letting the simulation output surface in narrative form to the player. This ties into managing information overload; as the simulation grows complex, presenting it in a user-friendly way is crucial. This area will involve UX design and maybe some web tech if we embed HTML for rich text and formatting.

- **Iteration and Playtesting:** With each new feature or system, continuous playtesting is vital. We should set up a pipeline for internal testing, and later closed alpha testing with players, to gather feedback. Particularly for an AI-driven game, observing how players interact and where they get frustrated or confused will guide a lot of tweaks. Perhaps players will come up with strategies or inputs we never anticipated – we must be ready to handle those either by improving AI understanding or adjusting game rules. Metrics to track during tests could include: average number of AI calls per minute (to gauge performance), how often the AI produces an invalid or nonsensical response (to improve prompt design), and qualitative logs of interesting emergent events (to see if our goals for narrative emergence are being met).

- **Team & Expertise Requirements:** As next steps, we might also identify key roles to bring on board. AI researchers or prompt engineers can help refine the AI's outputs and ensure it aligns with design goals. Game designers with experience in systemic games can help tune simulation parameters. Writers or narrative designers may be needed to craft the tone and voice of the AI-generated text (even if AI is writing, we guide it with examples). If seeking investors or partners, the next step is preparing demo materials (perhaps a video of the prototype demonstrating a compelling emergent storyline) to communicate the potential of Lagos Stories.

- **Long-Term Vision – "Holodeck" Ambition:** While focusing on immediate steps, we keep an eye on the long-term dream: a game that offers a **holodeck-like experience**, where the world is as adaptive as a real environment and the narrative possibilities are vast. Features like voice input/output (so players could speak to characters) or AR elements are far-off considerations. For now, getting the fundamentals right in a text-based medium is priority. But the design decisions we make should not preclude future expansion into richer media if desired.

In conclusion, the next steps are about validating the concept in a manageable setting, choosing the right tech stack (with Godot 4 and local AI inference being strong options), and then incrementally building complexity. By tackling risks early (AI integration, performance, player comprehension) and iterating, we aim to de-risk the innovative parts of the project. The journey will involve cutting-edge AI development and creative game design in tandem. With a successful prototype and clear plan, we can attract the necessary support (technical talent, partnerships, funding) to realize the full vision of Lagos Stories.

Lagos Stories – Project Brief (2025). Authored by Jason Ikeokwu.

Sources & Inspirations: *The design draws inspiration from emergent simulation games like Dwarf Fortress (open-ended sandbox design [2]) and The Sims, as well as recent advancements in AI-driven gameplay seen in projects like Retail Mage (AI as dynamic game master [1]) and systemic narrative experiments (City 20's emergent storytelling approach [3] [7]). These examples illustrate both the potential and challenges of combining AI with deep simulation, guiding our approach to Lagos Stories.*

---

[1] [5] [6] [8] [9] [10] Making Retail Mage: A New Approach to AI in Games — Jam & Tea Studios
https://www.jamandtea.studio/news/making-retail-mage-a-new-approach-to-ai-in-games

[2] Dwarf Fortress - Wikipedia
https://en.wikipedia.org/wiki/Dwarf_Fortress

[3] [7] "Civilization Without the Civility: Untold Games Reveals their latest title ‘City 20’, a Hugely Ambitious Speculative Survival Sim" - Games Press
https://www.gamespress.com/Civilization-Without-the-Civility-Untold-Games-Reveals-their-latest-ti

[4] Intra: design notes on an LLM-driven text adventure
https://ianbicking.org/blog/2025/07/intra-llm-text-adventure