**Procedural World Generation: Master Plan**

**1. High-Level Vision & Philosophy**

This document outlines the master plan for evolving the procedural world generation of *Cosmic Courier*. The current system serves as a functional placeholder, but the end goal is to create a deep, believable, and strategically interesting world that directly fuels the game's core loop as an "Epic Incremental / Logistics Simulation."

The generation will follow a top-down, hierarchical approach:

**World > Continent > Country/Region > City > District > Block**

Our core philosophy is to use a hybrid of procedural generation techniques, applying the best tool for each specific task. This creates a world that is not only visually appealing and organic but also functionally sound for a logistics-focused game.

* **Top-Down Structure (Noise & Pathfinding):** We will generate large-scale features like continents and major highways first. This provides a logical, consistent framework that guides the placement and shape of all smaller-scale details. The world's geography will dictate the flow of its infrastructure.
* **Rule-Based Systems (A\* & Suitability Mapping):** Cities and their primary road networks will be placed according to intelligent rules. This ensures that settlements appear in logical locations (e.g., on coasts, near rivers) and that the transportation network is efficient, creating meaningful logistical puzzles for the player to solve.
* **Aesthetic Detailing (Wave Function Collapse):** The final, high-fidelity details of city blocks—local streets, building variety, parks—will be generated using Wave Function Collapse (WFC). This will solve the "etch-a-sketch" problem of simpler algorithms and create cities that are dense, varied, and beautiful to observe.

**2. The Phased Action Plan**

To manage this complexity, we will follow a pragmatic "City First" development strategy. This allows us to build and integrate the most visually complex component (the city generator) first, providing immediate, tangible improvements to the playable game. This perfected city generator will then be "plugged into" the larger world once it's built.

**Phase 1: The City Sandbox - Building the Ideal City Generator**

**Goal:** Create a new, powerful, and standalone city generator that replaces the current system of walkers and simple districts.

* **1.1. Establish a "Generator Lab":** All new logic will be developed in a new, isolated service (services/NewCityGenService.lua). This prevents disruption to the currently playable game. The core of this service will be a single function: generateDetailedCity(params).
* **1.2. Define the params Contract:** This function will be designed from the start to accept a params table, which acts as its "contract" with the outside world. This is the key to easy integration later. The params will include everything a city needs to know about its environment, such as width, height, terrain\_data, and highway\_connections. In this initial phase, we will pass hardcoded placeholder values for testing.
* **1.3. Implement the "Blueprint" Logic:** This step generates the city's skeleton.
  + **Abstract Zoning:** Use a low-resolution WFC pass to generate organic "blobs" of influence for different zone types (downtown, industrial, etc.). This is the "districts as a thought" step, providing a high-level suggestion for the city's layout.
  + **Arterial Roads:** Use the A\* pathfinder (from lib/pathfinder.lua) to generate the city's main arteries. These roads will intelligently connect the highway\_connections from the params to the centers of the zone "blobs," ensuring a functional and efficient traffic flow.
* **1.4. Implement High-Fidelity Detailing:** This step fleshes out the skeleton.
  + **WFC for City Blocks:** Use a high-resolution WFC pass to fill in the areas *between* the arterial roads. This will generate all the local streets, building footprints, and small parks, creating organic and varied neighborhoods. This system will entirely replace the current connecting\_roads.lua walker logic.
* **1.5. Debug and Iterate:** A new debug option will be added to DebugMenuController.lua to call this new service and visualize the generated city, allowing for rapid iteration and refinement in isolation.

**Phase 2: Urban Renewal - Integrating the New City into the Game**

**Goal:** Replace the simplistic city generation in the current region view with the superior generator from Phase 1.

* **2.1. Perform the "Hotswap":** The city generation logic within services/MapGenerationService.lua will be replaced with a single call to NewCityGenService.generateDetailedCity(params).
* **2.2. Provide Placeholder params:** The params table passed to the new function will be hardcoded to match the context of the current, flat region map.
* **2.3. Stamp the Result:** The highly detailed city\_grid returned by the new function will be stamped onto the main region\_map.grid.
* **Outcome:** This will result in an immediate and dramatic visual and functional upgrade to the cities within the playable game, providing a much more engaging environment for testing core gameplay mechanics.

**Phase 3: The Top-Down World Builder**

**Goal:** With the city generator perfected, build the full-scale continent that will serve as the true game world.

* **3.1. Build the Continent Generator:** Implement the top-down generation logic: use noise functions to create a continental heightmap, define biomes (mountains, plains, water), and simulate logical river systems.
* **3.2. Generate Continental Infrastructure:** Use suitability mapping to find the best locations for cities. Then, use the A\* pathfinder to create a continental highway network that connects these city locations, respecting the terrain generated in the previous step.
* **3.3. Store World Data:** The output of this phase will be a persistent data file that defines the entire continent, its geography, and the specific parameters for each city location (its terrain, highway entry points, etc.).

**Phase 4: The Final Assembly**

**Goal:** Connect the finished systems by populating the continent from Phase 3 with the detailed cities from Phase 1.

* **4.1. The Master Orchestrator:** A final master function will load the persistent continent data.
* **4.2. Generate Cities in Place:** It will iterate through each city location on the continent. For each city, it will create a *real* params table based on the actual geography and highway locations at that point on the map.
* **4.3. The "Plug-in":** It will call the perfected NewCityGenService.generateDetailedCity() function, passing in this real, dynamically generated params table.
* **4.4. Final Integration:** The returned city grid will be stamped onto the continent map. Once all cities are generated, this new, complete world map will replace the old placeholder map, and the game's full vision for a dynamic, procedurally generated world will be realized.