# **Project Chimera: The Ultimate Comprehensive Development Plan**

## **Part 5: Core Gameplay Systems: Deep Dive (MVP & Full Vision)**

Version: 1.0

Date: May 27, 2025

Document Focus: Elaboration of Part 5 - Core Gameplay Systems: Deep Dive (MVP & Full Vision)

This section of the development plan transitions from the foundational Minimum Viable Product (MVP) to the fully realized, intricate gameplay systems that will define the complete Project Chimera experience. For each core system introduced in the MVP, we will first briefly summarize its MVP implementation and then provide an exhaustive deep dive into its "Full Vision." This includes detailing advanced mechanics, expanded features, increased simulation depth, new player interactions, C# implementation considerations for these advanced aspects, and potential AI tooling applications. The goal is to articulate the complete intended scope and complexity of each system, illustrating the evolutionary path from the MVP's core validation to the rich, multifaceted simulation of the final game.

### **5.1. Cultivation Mechanics: From Seed to Harvest (and Beyond)**

The cultivation system is the absolute heart of Project Chimera, simulating the intricate process of growing cannabis from a genetic starting point to a valuable harvested product. While the MVP establishes the manual basics, the full vision encompasses a far more nuanced, dynamic, and scientifically grounded simulation.

**5.1.1. Plant Lifecycle Simulation**

* **MVP Recap (as per Part 4, Sec 4.3.1):**
  + **Core Stages:** Seed/Clone, Germination, Seedling, Vegetative (simplified), Flowering (simplified), Harvestable.
  + **Manual Focus:** Player manually plants, waters, applies basic nutrients, and harvests.
  + **Visuals:** Basic visual changes per stage, rudimentary health indicators.
  + **Post-Harvest:** Simplified manual drying and curing with minimal quality impact.
* **Full Vision: Deep Lifecycle Simulation & Nuance:**
  + **Expanded Growth Stages & Sub-Phases:**
    - **Seed Viability & Germination:** Seeds will have a "viability" stat (potentially influenced by age, storage conditions, or parental genetics) affecting germination success rates. The germination process itself might involve choices (e.g., paper towel method, direct sow) with minor trade-offs. Temperature and moisture will be critical for germination.
    - **Seedling Stage (Detailed):** A vulnerable phase requiring precise, gentle environmental control (high humidity, moderate light, careful watering). Root development becomes a tracked (though perhaps abstracted) parameter.
    - **Vegetative Stage (Multi-Phase):**
      * **Early Veg (V1-V3 weeks conceptual):** Focus on establishing strong root systems and initial foliage. Sensitive to over/under watering and nutrient imbalances.
      * **Mid Veg (V4-V6 weeks):** Rapid foliage growth, stem thickening. Ideal time for topping, LST, and other training techniques. Increased nutrient and light demand.
      * **Late Veg (V7+ weeks or transition trigger):** Plant reaches desired size/structure before flipping to flower. May develop pre-flowers.
    - **Transition to Flower (Stretch Phase):** A distinct period (1-3 weeks) after switching the light cycle (e.g., to 12/12). Plants often exhibit rapid vertical growth ("stretch"). Nutrient needs begin to shift.
    - **Flowering Stage (Multi-Phase):**
      * **Early Flower (F1-F3 weeks):** Initial bud site development (pistil formation). Stretch may continue. Nutrient needs shift towards Phosphorus (P) and Potassium (K).
      * **Mid Flower (F4-F7 weeks):** Significant bud development and resin (trichome) production. Strong aroma development (if simulated). Peak P and K demand.
      * **Late Flower / Ripening (F8+ weeks):** Buds swell and mature. Trichome appearance changes (clear -> milky -> amber), a key indicator for harvest timing. Calyxes swell, pistils may change color and recede. Nutrient needs may decrease; some growers "flush" plants (provide plain water).
    - **Harvest Window:** A period where harvesting is optimal for desired effects/profiles. Harvesting too early or too late will impact cannabinoid/terpene profiles and overall quality.
    - **Senescence:** If left unharvested for too long, plants may begin to naturally degrade, with potential for hermaphroditism (in some strains under stress) or reduced quality.
  + **Dynamic Stage Durations:** While base durations will be genetically influenced (PlantStrainSO), the actual time spent in each stage will be dynamically affected by:
    - **GxE Interactions:** Optimal conditions accelerate progress through stages; stress can slow or stall it.
    - **Player Actions:** Certain training techniques or nutrient regimens might slightly alter stage timing.
    - **Strain Genetics:** Some strains will be inherently faster or slower flowering.
  + **Resource Consumption Curves:** Nutrient, water, and CO2 uptake will follow dynamic curves based on growth stage, plant size/biomass, and environmental conditions, rather than being simple fixed rates per stage.
  + **Advanced Plant Health Modeling:**
    - Beyond a single health stat, track sub-metrics like root health, leaf health, stem integrity.
    - Specific nutrient deficiencies/toxicities will have unique visual cues and physiological impacts (see 5.4).
    - Environmental stressors (heat, cold, low/high humidity, light stress) will have more nuanced impacts on specific growth processes and susceptibility to pests/diseases.
  + **Detailed Post-Harvest Processing (Full Vision - see Section 5.8):**
    - **Drying:** Critical environmental control (temperature, humidity, airflow) in the dry space directly impacts drying speed, terpene preservation, and risk of mold/over-drying. Different drying methods (hang drying, net drying) might have subtle effects.
    - **Trimming (Detailed):** Options for wet vs. dry trimming. The quality of the trim job (manual minigame or skill-based automated process) affects "bag appeal" and potentially final product categorization (e.g., A-grade buds, B-grade, trim for extracts).
    - **Curing (Detailed):** Long-term process in airtight containers. Critical management of humidity (RH) within jars (e.g., using hygrometers, Boveda-like humidity control packs as consumable items). Regular "burping" (gas exchange) is crucial. Proper curing significantly enhances flavor, aroma, smoothness, and potentially perceived potency. Different curing durations yield different results.
  + **C# Implementation (Full Vision):**
    - The PlantInstance.cs state machine for growth stages will become more granular, with more states and more complex transition logic.
    - GrowthStageSO.cs will need to store more detailed parameters for each sub-phase (e.g., ideal VPD, specific nutrient ratios, light DLI targets).
    - A PlantPhysiology.cs component within PlantInstance could manage resource uptake, transpiration, photosynthesis rates, and internal resource pools, all influenced by genetics and environment.
    - The PostHarvestController.cs will evolve into a more complex system managing drying room environments, trim quality calculations, and curing jar states (RH, burp cycles).
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the more complex state logic, physiological calculation methods, and data structures for detailed post-harvest tracking.
    - AI for Textures/Visuals: Generating more nuanced visual cues for specific nutrient deficiencies, stress responses, or trichome development stages for plant materials.

The full vision for the plant lifecycle aims for a deeply immersive and scientifically plausible simulation, where players must understand and manage each phase with care and precision to achieve optimal results.

**5.1.2. Genotype x Environment (GxE) Interaction: The Heart of Simulation**

* **MVP Recap (as per Part 4, Sec 4.2.2):**
  + Five foundational strains with distinct (but simple) optimal environmental "recipes."
  + Basic GxE model: manual environmental adjustments influence growth rate, health, and a few simple visual genetic traits (e.g., leaf shape, height).
  + Graduated plant responses to environmental closeness to optima.
* **Full Vision: Deep, Multifaceted GxE Simulation:**
  + **Comprehensive Trait Expression:** The GxE model will govern the expression of a wide array of complex, polygenic traits, including:
    - **Yield:** Final harvest weight (dry).
    - **Cannabinoid Profile:** Percentages of THC, CBD, CBG, CBN, THCV, etc.
    - **Terpene Profile:** Relative concentrations of key terpenes (Myrcene, Limonene, Pinene, Caryophyllene, Linalool, Humulene, etc.), which influence aroma, flavor, and potentially synergistic effects ("entourage effect").
    - **Growth Characteristics:** Internode length, branching patterns, leaf morphology, apical dominance, stretch factor.
    - **Flowering Time:** Actual duration of the flowering stage.
    - **Pest & Disease Resistance:** Modifying the plant's inherent genetic resistance based on environmental stress.
    - **Resin Production:** Density and maturity of trichomes.
    - **Visual Aesthetics:** Flower color, density, "bag appeal."
  + **Sophisticated Environmental Influence Modeling:**
    - **Environmental Factors:** Temperature (day/night differential, average, extremes), humidity (RH, VPD), light (intensity/PPFD, spectrum, DLI, photoperiod), CO2 concentration, airflow, growing medium properties (pH, EC, aeration, water retention - for soil/coco), nutrient solution parameters (for hydroponics).
    - **Response Curves (AnimationCurves):** Each genetically influenced trait (or sub-component of a trait) will have defined optimal ranges and response curves for key environmental factors. These curves, stored in PlantStrainSO or linked GxE\_ProfileSOs, will determine how far from optimal a plant is for that factor and apply a corresponding modifier (e.g., 0.0 to 1.5) to the trait's expression.
      * *Example:* A specific strain might have an optimal temperature for THC production of 25°C. At 20°C, its THC expression modifier might be 0.8; at 30°C, it might be 0.7 (due to heat stress).
    - **Cumulative Environmental Impact:** The GxE calculation will consider the *cumulative* environmental conditions over relevant growth stages, not just instantaneous values. For example, final yield is influenced by conditions throughout vegetative and flowering, while terpene profile might be more sensitive to late-flowering conditions. EnvironmentalSnapshot.cs (Doc3, Sec 2.2.2) will be crucial for tracking these aggregated historical values.
    - **Critical Thresholds & Stressors:** Exceeding critical environmental thresholds (e.g., extreme heat, frost, prolonged drought, severe nutrient toxicity) will trigger significant stress responses, potentially halting growth, damaging the plant, inducing hermaphroditism, or severely reducing quality/yield.
  + **Specific GxE Interaction Terms (Post-MVP Advanced):**
    - Beyond general environmental modifiers, specific gene-environment interactions can be modeled.
    - *Example:* A plant with a specific "mold resistance allele" might only express high resistance if humidity is kept below a certain threshold; above that, the allele's benefit diminishes. This could be modeled with lookup tables or conditional logic in the GxE calculation, referenced in GxE\_ProfileSOs.
  + **Nutrient GxE Interactions (Detailed):**
    - The availability and balance of specific macronutrients (N, P, K) and micronutrients at different growth stages will directly interact with genetic predispositions to influence specific traits.
    - *Example:* High P and K during flowering are crucial for bud development, but the *optimal ratio* and *uptake efficiency* might vary by strain genetics. Genetic traits for "Nutrient Uptake Efficiency" could be simulated.
  + **Light Spectrum GxE Interactions:** Different light spectrums (e.g., more blue light in veg, more red light in flower) can influence plant morphology (e.g., internode length, branching) and potentially secondary metabolite production (cannabinoids, terpenes). Certain strains might show more pronounced responses to specific spectrums. This would involve EquipmentDataSO for lights defining their spectral output, and PlantStrainSO having sensitivity factors.
  + **Data Presentation & Player Discovery:**
    - The UI will need to provide much more detailed feedback on environmental parameters and plant responses. Advanced graphs, historical logs, and potentially an "Environmental Stress Report" for each plant.
    - Players will discover optimal "environmental recipes" for their specific strains and genotypes through careful experimentation, observation, and data analysis, potentially aided by the "AI Research Lab" (post-MVP) which could simulate GxE outcomes.
  + **C# Implementation (Full Vision):**
    - GxE\_Calculator.cs: A sophisticated module responsible for taking a PlantStrainData (genetic potentials), an EnvironmentalSnapshot (aggregated historical environmental data), and potentially a GxE\_ProfileSO (defining specific interaction rules and response curves for that strain/genotype) to calculate the final expressed values for all key phenotypic traits.
    - EnvironmentalSnapshot.cs: Will need to be robust, tracking min/max/average values for many parameters over different time windows (e.g., per growth stage, last 24h, entire lifecycle).
    - PlantStrainSO.cs / GxE\_ProfileSO.cs: Will store numerous AnimationCurve assets for trait responses to various environmental factors, optimal ranges, and potentially data for specific interaction terms. Custom editors for these SOs will be essential for managing this complexity.
    - The core simulation loop will periodically update the EnvironmentalSnapshot for each plant, and the GxE\_Calculator will be invoked at key points (e.g., end of a growth stage, at harvest) to determine final trait expressions.
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the complex algorithms in GxE\_Calculator, data structures for EnvironmentalSnapshot, and custom editors for GxE profile SOs.
    - Unity Sentis (Post-MVP Research): Potentially running compact neural network models for highly complex, multi-variable GxE interactions that are difficult to model with explicit formulas or curves, as discussed in Part 3 (Sec 3.5.1). This would require training data from extensive game simulations.

The full GxE simulation is the scientific core of Project Chimera. Its depth and realism will be a major differentiator, offering endless possibilities for player experimentation and optimization in the pursuit of "ultimate cannabis genetics" expressed through masterful environmental control.

**5.1.3. Procedural Plant Morphology & Visual Realism (Full Vision)**

* **MVP Recap (as per Part 4, Sec 4.3.2 & 4.4.1):**
  + Five foundational strains with distinct visual archetypes (base models for procedural system).
  + Basic F1 crosses with simple Mendelian inheritance for a few visual traits (leaf shape, height, basic color).
  + Procedural generation system (code asset) and dynamic shaders enable basic visual changes reflecting GxE and simple genetics.
* **Full Vision: Highly Dynamic, Data-Driven Plant Visuals:**
  + **Goal:** Plants must be the "primary visual output of the GxE simulation" (Doc2, Sec V.B), dynamically and accurately reflecting their unique genetic makeup, growth stage, health status, environmental history, and any player-applied training techniques. No two plants (unless clones grown in identical conditions) should look exactly alike.
  + **Advanced Procedural Generation System (Hybrid Approach - Doc3, Sec 2.1.2):**
    - **Rule-Based Component System (Core Structure):**
      * Extensive library of high-quality, artist-created modular PlantPart prefabs (or data for procedural mesh generation): diverse leaf types (varying serration, width, length, leaflet count), stem segments (different thicknesses, textures), bud structures (cola shapes, calyx sizes, pistil variations), trichome models/shaders. These are the "building blocks."
      * PlantGenerationRuleSOs: Sophisticated ScriptableObjects defining rules for:
        + Branching patterns (angles, frequency, internode length).
        + Leaf placement, size, and orientation.
        + Bud site development, density, and morphology.
        + Overall plant architecture (apical dominance, bushiness, height-to-width ratio).
      * These rules will be heavily influenced by parameters derived from the plant's PlantStrainData (dozens of genetic factors influencing morphology) and its current EnvironmentalSnapshot (e.g., light availability affecting internode length and leaf size/orientation - phototropism).
    - **Spline-Based Structures:** Main stems and significant branches will be generated along dynamic splines. The control points and curvature of these splines will be influenced by genetics (e.g., "branch angle" genes), environmental factors (e.g., phototropism causing branches to bend towards light, gravitropism), and player training (LST physically manipulating these splines).
    - **L-Systems (for Fine Detail & Organic Variation - Optional Enhancement):**
      * Used for intricate details that are hard to achieve with purely component-based rules:
        + Leaflet arrangement and serration patterns on individual leaves.
        + Complex floret arrangements within buds.
        + Fine root structures (if ever visualized in detail, e.g., for hydroponics).
      * L-system parameters (axioms, production rules, angles, lengths) will be dynamically modulated by specific genetic traits and micro-environmental conditions.
  + **Dynamic Visual Responses to GxE & Health:**
    - **Leaf Color & Condition:** Shader-driven changes to leaf color and texture to represent:
      * Nutrient deficiencies/toxicities (e.g., yellowing/chlorosis for nitrogen deficiency, necrotic spots for calcium deficiency, tip burn for nutrient excess).
      * Water stress (wilting, drooping, crispy edges).
      * Light stress (bleaching, burning).
      * Disease symptoms (powdery mildew shaders, leaf spot textures).
      * Pest damage (stippling from mites, chewed edges).
    - **Leaf Turgidity & Orientation:** Leaves dynamically adjust their angle and "perkiness" based on hydration status and light exposure (phototropism).
    - **Stem Thickness & Color:** Stems thicken with age and biomass accumulation. Color might change based on strain or stress.
    - **Bud Development:** Visual progression of bud swelling, calyx development, pistil color changes (white -> orange/brown), and trichome maturation (clear -> milky -> amber, visualized through shader effects or texture swaps on trichome models/layers).
    - **Overall Vigor:** Healthy, optimally grown plants appear lush and vibrant. Stressed or unhealthy plants look stunted, sparse, or discolored.
  + **Impact of Player Training Techniques:**
    - **Topping/Fimming:** Procedural system responds by altering branching rules, typically creating multiple main colas from the cut point.
    - **Low-Stress Training (LST):** Player actions of bending and tying down branches directly manipulate the control splines of those branches, influencing their growth direction and light exposure, which in turn affects bud development along those branches.
    - **Defoliation:** Removing leaves updates the plant model and can affect light penetration to lower bud sites, potentially influencing their development (a GxE factor).
    - **Super Cropping (Advanced):** If implemented, crushing a stem interior would trigger a "knuckle" formation and potentially altered growth above that point.
  + **Genetic Visual Differentiation:**
    - Beyond the MVP's simple visual traits, the full system will allow for a vast spectrum of visual differences based on complex genetic combinations:
      * Overall plant structure (tall & lanky Sativa-like, short & bushy Indica-like, diverse hybrid forms).
      * Leaf morphology (number of leaflets, width, serration patterns).
      * Flower coloration (greens, purples, reds, oranges – influenced by anthocyanin genetics and temperature).
      * Bud density and structure (airy vs. dense, foxtailing).
      * Trichome coverage and appearance.
  + **Performance Optimization (CRITICAL for many dynamic plants):**
    - **Aggressive LODs:** Essential for all plant parts and the assembled plant.
    - **MaterialPropertyBlocks:** To change material properties (e.g., color, texture offsets for stress effects) per plant instance without creating unique materials, which breaks batching.
    - **Shader Optimization:** Highly optimized shaders for plant rendering.
    - **Impostors/Billboards:** For very distant plants, potentially use impostors or simplified billboards.
    - **Selective Updates:** Only update the visual mesh/shaders of plants when their state significantly changes or when they are visible.
    - **Object Pooling:** For plant parts if they are frequently swapped or regenerated.
  + **C# Implementation (Full Vision):**
    - ProceduralPlantGenerator.cs: A sophisticated manager class responsible for constructing and updating the visual representation of each PlantInstance. It takes PlantStrainData, EnvironmentalSnapshot, current growth stage, and health status as inputs.
    - It will contain the logic for rule-based assembly, spline manipulation, and potentially invoking L-system generation modules.
    - PlantVisualsController.cs (on PlantInstance): Interfaces with ProceduralPlantGenerator and manages the plant's specific mesh renderers, materials, and shader parameters.
    - A library of PlantPartSOs or similar data structures defining the properties and rules for each modular plant component.
    - Complex C# scripts for dynamic shader control, responding to health and GxE data.
  + **AI Tooling (Full Vision):**
    - AI for 3D Assets (Rodin, etc.): Generating the high-quality base meshes for the diverse library of PlantParts (leaves, stems, buds, trichomes) that the procedural system will use.
    - AI for Textures (Stable Diffusion, Substance Sampler): Creating the detailed PBR base textures and variation maps (e.g., for different deficiency symptoms, color morphs) for these plant parts.
    - Unity AI (6.2 Beta - Sentis/Generators - Experimental): Potentially explore if new Unity AI tools can assist in runtime procedural mesh deformation, texture blending, or even generating subtle variations in plant part geometry based on simulation data, beyond what the rule-based system does. This is an R&D area.

The full vision for procedural plant morphology aims to create a stunningly realistic and dynamic visual experience, where every plant tells a story of its unique genetics and the journey orchestrated by the player and the environment. This is a highly complex technical and artistic challenge but is central to Project Chimera's appeal.

**5.1.4. Player Cultivation Techniques (Manual & Automated)**

* **MVP Recap (as per Part 4, Sec 4.3.1):**
  + Manual planting, watering, basic nutrient application.
  + Optional, highly simplified pruning/topping.
  + No automation beyond simple light timers.
* **Full Vision: A Rich Toolkit of Techniques & Gradual Automation:**
  + **Advanced Manual Training Techniques:**
    - **Low-Stress Training (LST):** Players can physically (via UI interaction) bend and tie down branches. This directly manipulates the plant's procedural generation splines, altering growth direction to create a more even canopy, improve light exposure to lower sites, and potentially increase the number of main colas. Requires LST clips/ties as conceptual items or just a tool mode.
    - **High-Stress Training (HST):**
      * **Topping/Fimming:** More precise control over where to cut the apical meristem, with predictable (multiple colas) vs. less predictable (FIM - "Fuck I Missed" - potentially more, smaller colas) outcomes.
      * **Super Cropping:** Damaging the inner tissue of a stem to create a "knuckle" and encourage lateral growth or redirect vertical growth.
      * **Lollipopping:** Removing lower branches and foliage that receive little light to focus energy on upper colas.
    - **Screen of Green (ScrOG):** Placing a screen/net over plants and training branches to grow horizontally along it, creating a flat, wide canopy of bud sites. Requires a placeable "ScrOG Net" equipment item.
    - **Sea of Green (SOG):** Growing many small plants close together, focusing on a single main cola per plant, to achieve a quick harvest cycle. More of a strategy than a direct technique on one plant, but supported by allowing dense planting.
    - **Defoliation (Strategic):** Selective removal of fan leaves at specific times (e.g., late veg, mid-flower) to improve light penetration and airflow to bud sites. Over-defoliation can stress the plant. This would be a skill-based action, perhaps with a visual overlay showing "safe" leaves to remove.
  + **Automated Systems (Post-MVP - "Earned Automation"):**
    - **Automated Irrigation/Fertigation:**
      * Drip systems, ebb & flow tables, deep water culture (DWC) / recirculating deep water culture (RDWC) systems. These are placeable equipment types.
      * Requires connection to water/nutrient reservoirs via the plumbing utility network (see 5.3).
      * Can be controlled by timers (for scheduled watering/feeding) or advanced controllers linked to sensors (e.g., soil moisture sensors, reservoir EC/pH sensors).
    - **Automated Environmental Control (see 5.3):** Advanced sensors and controllers managing lights, temperature, humidity, CO2, and airflow based on player-defined recipes or dynamic plant needs.
    - **Automated Nutrient Dosing:** Systems that automatically mix and adjust nutrient solutions in reservoirs based on sensor readings (EC/pH) and target recipes. Requires doser pump equipment.
    - **Workflow Automation (Late-Game - Doc1, Sec VII.C):**
      * **Automated Potting/Transplanting Machines:** For large-scale operations.
      * **Robotic Harvesting/Trimming Systems:** Visual/functional representation of machines that can automate parts of the harvest and trimming process, likely with trade-offs in quality vs. speed/labor cost.
      * **Automated Plant Movement:** Conveyor belts or robotic platforms for moving plants between different grow zones (e.g., veg room to flower room).
  + **Hydroponic & Aeroponic Systems (Advanced Cultivation Methods):**
    - Placeable equipment for DWC, RDWC, Nutrient Film Technique (NFT), aeroponics.
    - These systems have different resource requirements (e.g., more power for pumps, chillers for reservoirs), different risks (e.g., rapid spread of root diseases if water is contaminated), and potentially different growth rates or GxE responses compared to soil/coco.
    - Requires careful management of nutrient solution temperature, oxygenation, EC, and pH.
  + **Living Soil & Organic Cultivation (Advanced Niche):**
    - Option to create "living soil" mixes with beneficial microbes, compost teas, and organic amendments.
    - More complex soil food web simulation (abstracted). Focus on building soil health over time rather than just applying synthetic nutrients.
    - May offer unique quality benefits or challenges.
  + **C# Implementation (Full Vision):**
    - PlayerInteractionController.cs will need modes for different training techniques (LST spline manipulation, targeting for topping/defoliation).
    - PlantInstance.cs will have methods to respond to these training actions, altering its procedural generation parameters or internal growth logic.
    - New equipment classes for all automated systems (AutomatedDripper.cs, HydroponicBasinController.cs, RoboticArm.cs). These will interact heavily with the EnvironmentController, ResourceManager, and potentially a central AutomationManager.cs.
    - SoilSystem.cs / HydroponicSystem.cs: More detailed simulation of the growing medium/solution and its interaction with plant roots.
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the C# logic for the diverse equipment controllers, state management for automated processes, and the more complex interactions between training techniques and plant procedural generation.

The full vision for cultivation techniques provides players with a vast toolbox of methods, from intricate manual training to sophisticated automation, allowing them to tailor their approach to different strains, scales of operation, and personal playstyles. This progression from fully manual to highly automated is a core part of the long-term player journey.

**5.1.4. Player Cultivation Techniques (Manual & Automated) - Full Vision Continued**

* **MVP Recap (as per Part 4, Sec 4.3.1):**
  + Manual planting, watering, basic nutrient application.
  + Optional, highly simplified pruning/topping.
  + No automation beyond simple light timers.
* **Full Vision: A Rich Toolkit of Techniques & Gradual Automation (Continued from previous response):**
  + **Advanced Manual Training Techniques (Detailed Implementation):**
    - **Low-Stress Training (LST):**
      * **Player Interaction:** The player would select an "LST Tool" or enter an "LST Mode." When a young, pliable branch is selected, the player could click and drag a point on the branch (or its representative spline) to a new anchor point on the pot rim, another branch, or a dedicated LST anchor point item. A visual representation of a "tie" would appear.
      * **Procedural Impact:** This action directly modifies the control points of the B-spline or Bezier curve representing that branch in the ProceduralPlantGenerator. The plant's growth simulation would then continue along this new curve. Over time, the branch "hardens" in its new position.
      * **GxE Feedback:** The altered branch position changes its light exposure and potentially airflow around it. The procedural system might trigger increased lateral bud development along the now-horizontal branch due to altered apical dominance (simulated effect) and better light.
      * **C#:** LSTController.cs handling input and spline manipulation. PlantInstance.cs would store LST anchor points and apply tension modifiers to its growth simulation. ProceduralPlantGenerator.cs reads these spline modifications.
    - **Topping/Fimming (HST):**
      * **Player Interaction:** Player selects a "Pruning Tool" (e.g., sterile scalpel/snips item). When hovering over the apical meristem (main growing tip) of a young plant, a UI prompt appears. "Topping" removes the tip cleanly. "Fimming" (a more imprecise cut) might be a variant action with a slightly different outcome.
      * **Procedural Impact:**
        + Topping: The ProceduralPlantGenerator detects the "topped" status. The growth rule for the apical meristem is halted. Rules for the two nodes immediately below the cut are activated to develop into new dominant colas.
        + Fimming: Might result in 2-4 smaller, less dominant colas from the cut area due to a less clean removal of the apical tip, simulating a "bushier" but potentially less organized top.
      * **GxE Feedback:** Plant might experience a brief period of slowed growth (stress response) before redirecting energy to the new colas.
      * **C#:** PruningController.cs with modes for topping/fimming. PlantInstance.cs flags the apical meristem as topped/fimmed. ProceduralPlantGenerator.cs implements specific branching logic post-cut.
    - **Super Cropping (HST):**
      * **Player Interaction:** Select a "Stem Crushing" action or tool. Player targets a point on a pliable (but not too young) stem. A brief animation/sound effect occurs.
      * **Procedural Impact:** The ProceduralPlantGenerator creates a visual "knuckle" (thickened area) at that point on the stem model over a few in-game days. The stem segment above the crop might initially droop then recover, growing with a more horizontal orientation before turning upwards again. Growth hormones (auxins) are conceptually redirected, potentially boosting growth in branches below the crop.
      * **GxE Feedback:** A temporary stress response, followed by potentially increased vigor in lower branches and altered canopy structure. Risk of stem snapping if done incorrectly or on a brittle stem (small chance).
      * **C#:** TrainingController.cs handles the super cropping action. PlantInstance.cs flags a stem segment as super-cropped, influencing its growth angle and potentially hormone distribution logic.
    - **Lollipopping & Strategic Defoliation:**
      * **Player Interaction:** Using the "Pruning Tool," player selects individual fan leaves or small lower branches for removal. A UI overlay might highlight "safe" or "recommended" leaves/branches to remove based on plant stage and light exposure simulation.
      * **Procedural Impact:** Selected leaves/branches are visually removed from the plant model by the ProceduralPlantGenerator.
      * **GxE Feedback:**
        + Improved light penetration to lower/inner bud sites, potentially increasing their size and density.
        + Improved airflow through the canopy, potentially reducing humidity locally and lowering risk of mold.
        + Over-defoliation causes significant stress, reducing photosynthesis capacity and potentially stunting growth or reducing yield/quality. A "Leaf Area Index" (LAI) concept might be simulated abstractly.
      * **C#:** DefoliationController.cs manages selection and removal. PlantInstance.cs tracks its LAI or photosynthetic capacity. ProceduralPlantGenerator.cs updates visuals. Light simulation recalculates penetration.
    - **Screen of Green (ScrOG):**
      * **Player Interaction:** Player purchases and places a "ScrOG Net" equipment item over their plants during vegetative growth. As branches grow, the player uses LST-like interactions to weave and tuck branches through the net openings, training them to grow horizontally beneath the screen.
      * **Procedural Impact:** The ProceduralPlantGenerator respects the "ceiling" of the ScrOG net. Branches hitting the net are guided horizontally. Bud sites develop upwards from the horizontal branches, creating an even canopy of colas at the net level.
      * **GxE Feedback:** Maximizes light exposure to a large number of bud sites. Requires diligent, ongoing training by the player.
      * **C#:** ScrOGNet.cs (equipment) defines the grid. LSTController.cs interacts with branches and the net. ProceduralPlantGenerator.cs implements growth logic constrained by the net.
    - **Sea of Green (SOG):**
      * **Player Interaction:** This is more a planting strategy. Player densely plants many clones or small seedlings and quickly flips them to flower with minimal vegetative time. The goal is one main cola per plant.
      * **Procedural Impact:** Individual plants remain small, focusing energy on a single apical bud. The ProceduralPlantGenerator emphasizes vertical growth and central cola development for plants under an SOG light cycle/density regime.
      * **GxE Feedback:** Shorter overall cycle time. Requires managing many small plants. Risk of overcrowding if not managed well.
      * **C#:** No specific new technique controller, but the core plant growth and density simulation must support this strategy effectively.
  + **Automated Systems (Post-MVP - "Earned Automation" - Detailed Implementation):**
    - **Automated Irrigation/Fertigation Systems:**
      * **Equipment:** DripEmitterSO, EbbFlowTableSO, DWCBucketSO, RDWCPipeSO, WaterPumpSO, NutrientReservoirSO. These are placeable equipment with costs, power needs, capacities, and connection points for the plumbing utility network.
      * **Logic:**
        + IrrigationController.cs (can be part of AutomationManager.cs or a dedicated system): Manages irrigation schedules or sensor-based triggers.
        + Player UI: Configure schedules (e.g., "Water for 5 mins every 4 hours") or set sensor thresholds (e.g., "Activate drip when soil moisture sensor in Pot X reads < 30%").
        + The system draws water/nutrient solution from connected reservoirs, pumps it through the network, and delivers it to plants. Flow rates, pressure (abstracted), and pipe capacity in the plumbing network (see 5.3) affect delivery efficiency.
      * **C#:** Reservoir.cs tracks fluid level and composition. Pump.cs moves fluid. PlantContainer.cs (pot/hydro bucket) receives fluid and updates its medium's moisture/nutrient levels. SoilMoistureSensor.cs provides data.
    - **Automated Nutrient Dosing Systems:**
      * **Equipment:** DosingPumpSO (peristaltic pumps for precise additions of concentrated nutrients/pH adjusters), InlineECSensorSO, InlinepHSensorSO, CentralNutrientControllerSO.
      * **Logic:**
        + NutrientDosingController.cs: Player defines a target EC/pH profile for a reservoir.
        + Inline sensors continuously monitor the reservoir solution.
        + If readings deviate from target, the controller activates specific dosing pumps to add small amounts of concentrated nutrient stock solutions (e.g., "pH Up," "pH Down," "Base A," "Base B," specific additives) to bring the solution back into range.
        + Requires careful calibration by the player (or through research unlocks) to prevent overshooting targets.
      * **C#:** Complex feedback loop logic. DosingPump.cs adds specific amounts of NutrientConcentrateSO to a Reservoir.cs. InlineSensor.cs provides real-time data.
    - **Workflow Automation (Late-Game - Very Advanced):**
      * **Automated Potting/Transplanting Machine:** An expensive, large piece of equipment. Player loads it with empty pots, growing medium, and seedlings/clones. The machine (via animation and abstracted process) "pots" them. Requires power and potentially maintenance.
      * **Robotic Harvesting/Trimming Systems:** Large, complex machines. Player feeds mature plants into one end. The machine performs harvesting and (a rougher) trim. Output is harvested buds (perhaps of slightly lower quality than perfect hand-trimming) and trim waste. High power consumption, high cost, requires research.
      * **Automated Plant Movement (Conveyors/Robots):** For massive facilities. Placeable conveyor belts or programmable robotic arms that can move potted plants between different grow rooms (e.g., from a dedicated seedling/clone room to a veg room, then to a flower room). Requires complex pathfinding (for robots) or layout planning (for conveyors) and a central LogisticsManager.cs.
      * **C#:** Each of these would be a major system. AutomatedPottingMachine.cs, RoboticHarvester.cs, ConveyorBelt.cs, RoboticPlantMover.cs. They would have internal state machines, resource requirements, and interact with PlantInstance and InventoryManager.
  + **Hydroponic & Aeroponic Systems (Advanced Cultivation Methods - Detailed Implementation):**
    - **Deep Water Culture (DWC) / Recirculating Deep Water Culture (RDWC):**
      * **Equipment:** DWCBucketSO, AirPumpSO, AirStoneSO, WaterChillerSO (for RDWC/larger DWC), RDWCReturnPumpSO, connecting pipes.
      * **Simulation:** Plant roots are submerged in highly oxygenated nutrient solution.
        + **Oxygenation:** Critical. Air pump + air stone simulation (buff to root health/nutrient uptake if active).
        + **Solution Temperature:** Must be kept within optimal range (e.g., 18-22°C) via Water Chillers, as warmer water holds less dissolved oxygen and promotes pathogens. WaterChiller.cs consumes power and cools solution in connected reservoirs.
        + **Nutrient Management:** EC/pH of the entire system (for RDWC) or individual buckets (for DWC) must be meticulously managed. Prone to rapid fluctuations.
        + **Root Health:** Highly visible roots. Susceptible to root rot (Pythium, etc.) if oxygen is low, temps are high, or solution is contaminated. RootHealth becomes a critical sub-stat of PlantHealth.
    - **Nutrient Film Technique (NFT):**
      * **Equipment:** NFTChannelSO, ShallowReservoirSO, FeedPumpSO.
      * **Simulation:** A shallow stream of nutrient solution flows over the bare roots of plants in channels.
        + **Flow Rate:** Critical. Too slow = poor oxygenation/nutrient delivery. Too fast = root damage.
        + Requires precise channel slope (abstracted or player-set).
    - **Aeroponics:**
      * **Equipment:** AeroponicChamberSO, HighPressurePumpSO, FineMistNozzleSO.
      * **Simulation:** Plant roots are suspended in air within a chamber and periodically sprayed with a fine mist of nutrient solution.
        + **Misting Cycle:** Timer-controlled (e.g., 15 seconds on, 5 minutes off). Nozzle clogging is a potential malfunction.
        + Highest oxygenation but least forgiving of pump failure or clogged nozzles (roots dry out quickly).
    - **C# (Hydro/Aero):**
      * New derived classes from BaseCultivationSystem.cs (e.g., DWCBucketSystem.cs, NFTChannelSystem.cs).
      * These systems will have unique parameters for PlantInstance GxE calculations (e.g., different nutrient uptake efficiency modifiers, different susceptibility to root zone issues).
      * RootZoneManager.cs (or similar) to simulate dissolved oxygen, solution temperature, pathogen risk specifically for hydro/aero systems.
  + **Living Soil & Organic Cultivation (Advanced Niche - Detailed Implementation):**
    - **Concept:** Focus on building a healthy soil ecosystem that provides nutrients naturally, rather than relying on synthetic bottled nutrients.
    - **Mechanics:**
      * **Soil Mixing:** Players can create custom soil mixes using base components (PeatMossSO, CompostSO, PerliteSO, WormCastingsSO, etc.) and organic amendments (KelpMealSO, BoneMealSO, BatGuanoSO). Each ingredient contributes to soil structure, water retention, aeration, and slow-release nutrient content. A SoilMixingUI.cs would allow this.
      * **Soil Food Web (Abstracted):** A SoilHealth stat for each pot of living soil. This stat is influenced by:
        + Initial mix quality.
        + Application of CompostTeaSO (consumable, brewed by player in a CompostTeaBrewerSO equipment).
        + Use of organic top dressings or amendments over time.
        + Avoiding synthetic pesticides/fungicides that harm microbial life.
      * **Nutrient Cycling:** Nutrients are slowly released as organic matter decomposes (simulated by SoilHealth influencing nutrient availability to the plant). Less direct control over PPMs than synthetics.
      * **"No-Till" / Re-amending:** Option to re-amend used soil for subsequent grows, potentially improving its quality over time if managed well.
      * **Benefits/Trade-offs:** May lead to enhanced terpene profiles or unique qualities ("organic" tag for products). Can be more challenging to diagnose/correct nutrient issues quickly. Potentially higher resistance to some soil-borne diseases if SoilHealth is high.
    - **C#:**
      * LivingSoilPot.cs (derived from PotContainer.cs): Tracks SoilHealth, current available nutrients (released by simulation), microbial activity level (abstracted).
      * OrganicAmendmentSO.cs: Defines nutrient content and release rate.
      * SoilFoodWebSimulator.cs: Module that updates SoilHealth and nutrient release based on amendments, watering, and time.
  + **Overall C# Implementation (Full Vision for 5.1.4):**
    - A highly modular system with clear interfaces for different training tools (ITrainingTool), automation controllers (IAutomationController), and cultivation system types (ICultivationSystem).
    - PlantInstance.cs will need to be very flexible to respond to inputs from all these diverse techniques and systems. Its UpdateGrowthAndHealth() method will become a complex orchestrator of genetic expression, environmental response, and effects of player interventions.
    - Event-driven architecture will be crucial (e.g., OnLSTAppliedEvent, OnAutomatedWateringCycleCompleteEvent, OnSoilHealthChangedEvent).
  + **AI Tooling (Full Vision for 5.1.4):**
    - Cursor AI: Indispensable for generating the boilerplate and core logic for the numerous new equipment controller classes, the state machines for automated processes, the complex interaction logic for advanced training techniques, and the data structures for new SO types (amendments, hydro components).
    - Unity AI Assistant: Useful for quick API lookups or simple snippets when implementing the C# logic for these diverse systems within the Unity editor.
    - AI for Concept Art/Reference Images: Generating visual ideas for the advanced equipment (robotic arms, hydroponic setups, organic tea brewers).

The full vision for Player Cultivation Techniques transforms Project Chimera into an incredibly deep sandbox, allowing players to explore a vast range of horticultural practices, from meticulous hands-on artistry to large-scale industrial automation and specialized organic or hydroponic approaches. This provides immense replayability and caters to diverse player interests and preferred playstyles.

### 5.2. Genetics & Breeding: The Pursuit of Perfection

The genetics and breeding system is arguably Project Chimera's most defining pillar, aiming to offer an unparalleled simulation of cannabis genetic engineering. While the MVP introduces the very basics, the full vision is to empower players with the tools and knowledge to become master breeders, capable of creating, refining, and stabilizing unique and high-performance cannabis strains. This involves a deep dive into polygenic traits, complex inheritance models, advanced breeding techniques, and sophisticated analytical tools.

* **MVP Recap (as per Part 4, Sec 4.3.2):**
  + **Content:** Five foundational strains with a few simple, observable visual genetic traits (e.g., leaf shape, height, basic flower color). Underlying data structures for full genetic complexity exist but are not actively used for complex traits in MVP.
  + **Mechanics:** Basic F1 crosses via manual pollination. Simple Mendelian inheritance for the visual MVP traits.
  + **Tools:** Rudimentary "Trait Library" for discovered visual traits. Streamlined "Genetics Lab UI" for parent selection and F1 seed creation. Basic seed bank.
  + **Exclusions:** No polygenic trait inheritance for yield/potency/terpenes, no advanced breeding techniques (BX, IBLs), no AI Research Lab, no mutations, no tissue culture.
* **Full Vision: Deep, Sophisticated Genetic Simulation & Strategic Breeding:**

**5.2.1. Genetic Representation:** Polygenic Traits, Genes, Alleles (Full Detail)  
The full vision requires a robust and nuanced system for representing the genetic makeup of each plant, moving far beyond the MVP's simple visual traits to encompass the complex quantitative traits that define a strain's value and characteristics.

* + **Polygenic Trait Modeling (Comprehensive List):**
    - **Primary Cannabinoids:** THC (Tetrahydrocannabinol), CBD (Cannabidiol), CBG (Cannabigerol) – each treated as a quantitative trait influenced by multiple genes controlling precursor production, synthase enzyme activity, and conversion pathways. Potentials for THCA, CBDA, CBGA (acidic forms) will be the primary genetic drivers, with decarboxylation being a post-harvest factor.
    - **Minor Cannabinoids:** CBN (Cannabinol - degradation product, but genetics might influence THC stability), THCV (Tetrahydrocannabivarin), CBDV (Cannabidivarin), CBC (Cannabichromene) – each with their own set of influencing genes.
    - **Terpene Profiles (Key Differentiator):** Individual concentrations of numerous key terpenes:
      * Monoterpenes: Myrcene, Pinene (alpha & beta), Limonene, Linalool, Terpinolene, Ocimene.
      * Sesquiterpenes: Caryophyllene (beta), Humulene, Bisabolol.
      * Each terpene's production will be a polygenic trait, influenced by genes controlling specific terpene synthase enzymes and precursor pathways. The overall "aroma profile" emerges from the combination of these.
    - **Yield & Biomass:** Overall dry flower yield, total biomass, potentially harvest index (ratio of bud weight to total plant weight). Influenced by genes for photosynthetic efficiency, nutrient uptake/utilization, plant structure, and stress tolerance.
    - **Growth Characteristics:**
      * Flowering Time (Photoperiod & Autoflower): Days to mature after initiating flowering. Separate genetic pathways for photoperiod-dependent flowering and autoflowering traits (e.g., linkage to *Cannabis ruderalis* genes).
      * Plant Height & Stretch: Overall height, internode length, degree of "stretch" during early flowering.
      * Branching Pattern & Apical Dominance: Bushy vs. columnar structure, number and angle of lateral branches.
      * Leaf Morphology: Size, width, number of leaflets, serration patterns (beyond MVP's simple categories).
      * Root System Development: Vigor and density of root growth (abstracted, but influences nutrient/water uptake).
    - **Pest & Disease Resistance:** Genetic predisposition to resist specific common pests (e.g., spider mites, aphids) and diseases (e.g., powdery mildew, botrytis/bud rot, root rot). Likely modeled as threshold traits or quantitative resistance scores.
    - **Stress Tolerance:** Genetic factors influencing tolerance to environmental stressors like heat, cold, drought, high/low humidity, nutrient imbalances.
    - **Aesthetic Traits:** Flower color (anthocyanin production influenced by genetics and temperature), bud density, trichome density/appearance ("frostiness"), overall "bag appeal" (an abstracted score).
    - **Seed Characteristics:** Seed size, viability, dormancy period.
  + **Gene & Allele Structure (Detailed - building on Doc3, Sec 2.2.1):**
    - **GeneDefinitionSO:**
      * geneID (unique string/int).
      * geneName (e.g., "THCA Synthase Gene Alpha").
      * description (functional role).
      * chromosomalLocus (string or int - for future genetic mapping, optional for initial full vision but good to have).
      * traitInfluences (List<TraitEffectEntry>): Specifies which phenotypic traits this gene contributes to. Each entry links to a PhenotypicTraitSO (see below) and describes the *nature* of its influence (e.g., "primary controller," "minor modifier").
      * possibleAlleles (List<AlleleDefinitionSO>): A list of all known alleles for this gene.
      * mutationRateModifier (float): Some genes might be more prone to mutation.
    - **AlleleDefinitionSO:**
      * alleleID (unique string/int).
      * alleleSymbol (e.g., "THCA\_High," "thca\_low," "PM\_Res\_R1," "pm\_res\_r1").
      * description (e.g., "High activity THCA synthase enzyme variant").
      * dominanceRelationship (enum: Dominant, Recessive, CoDominant, IncompleteDominant - relative to other alleles for *this specific gene* if simple dominance is modeled for some qualitative effects, though most traits are polygenic).
      * quantitativeEffectValues (Dictionary<PhenotypicTraitSO, float>): For each trait this allele influences, the additive (or multiplicative, if designed that way) effect value it contributes. E.g., THCA\_High might contribute +2.0 to a "THC Potential Score," while thca\_low contributes +0.5. These values are summed across all relevant genes to get the total genetic potential for a polygenic trait.
      * interactionTriggers (List<AlleleInteractionRuleSO>): For advanced epistasis/pleiotropy, this could link to rules defining how this allele interacts with alleles at *other* gene loci.
    - **PhenotypicTraitSO:**
      * traitID (unique string/int, e.g., "MaxTHCPotential," "MyrceneLevel," "FloweringTimeDays").
      * traitName (user-facing name).
      * description.
      * unitOfMeasurement (e.g., "%", "g/plant", "days", "ppm").
      * baseValue (float): A species or general baseline value for this trait.
      * minPossibleValue, maxPossibleValue (for clamping and UI).
      * isVisualMVPTrait (bool): Flag if it's one of the simple visual traits from MVP.
  + **PlantGenomeData (Runtime Class/Struct for each plant instance):**
    - This structure holds the complete diploid set of alleles for an individual plant.
    - **Representation:**
      * Dictionary<GeneDefinitionSO, AllelePairStruct> where AllelePairStruct holds two AlleleDefinitionSO references (one paternal, one maternal). This is clear but can be memory-intensive for thousands of genes if plants are stored long-term.
      * **Optimized Alternative:** Dictionary<GeneDefinitionSO, int[2]> where the int[2] stores the indices or unique IDs of the two alleles within the GeneDefinitionSO.possibleAlleles list. This is more compact.
      * Or, for extreme optimization if profiling demands it, parallel arrays: GeneDefinitionSO[] allGenes; AlleleDefinitionSO[] paternalAlleles; AlleleDefinitionSO[] maternalAlleles; where arrays are indexed consistently.
    - This PlantGenomeData is what is passed from parent to offspring during breeding and is a core part of a plant's persistent data.
  + **Calculating Genetic Potential for Polygenic Traits:**
    - For each PhenotypicTraitSO:
      * Initialize currentPotential = traitSO.baseValue.
      * Iterate through all GeneDefinitionSOs that list this PhenotypicTraitSO in their traitInfluences.
      * For each such gene, get the plant's two AlleleDefinitionSOs from its PlantGenomeData.
      * Add the quantitativeEffectValues for this trait from both alleles to currentPotential.
      * finalGeneticPotential = Clamp(currentPotential, traitSO.minPossibleValue, traitSO.maxPossibleValue).
    - This finalGeneticPotential is then what feeds into the GxE calculations (Section 5.1.2) to determine the actual expressed phenotype.

This detailed genetic representation allows for immense combinatorial possibilities, ensuring that breeding is a deep and rewarding system where players are truly exploring a vast genetic landscape.

**5.2.2. Inheritance Models:** From Simple Mendelian to Complex Interactions (Full Vision)The full vision moves beyond MVP's simple Mendelian inheritance for visual traits to simulate more realistic and complex genetic interactions for the numerous polygenic traits.

* + **Core Breeding Logic (Segregation & Independent Assortment):**
    - When two parent plants (PlantGenomeData P1, PlantGenomeData P2) are bred:
    - For each GeneDefinitionSO in the species' gene pool:
      * The offspring receives one allele for that gene from P1 (randomly selected from P1's two alleles for that gene) and one allele from P2 (randomly selected from P2's two alleles for that gene). This simulates Mendelian segregation and independent assortment (assuming genes are on different chromosomes or far apart, for simplicity, though linkage could be an advanced future layer).
    - This process creates the new diploid PlantGenomeData for the F1 offspring.
  + **Dominance Relationships (for Qualitative or Threshold Effects):**
    - While most key traits are polygenic and additive, some specific effects or visual markers might still follow simpler dominance/recessive patterns if desired for variety (e.g., a specific rare flower color mutation).
    - The AlleleDefinitionSO.dominanceRelationship field would be used here, in conjunction with the paired allele, to determine the expressed phenotype for that specific simple trait.
  + **Co-dominance & Incomplete Dominance (for specific traits):**
    - **Co-dominance:** Both alleles are expressed fully and distinctly (e.g., a flower with patches of two different colors if color alleles were co-dominant).
    - **Incomplete Dominance:** The heterozygous phenotype is an intermediate blend of the two homozygous phenotypes (e.g., Red (RR) x White (rr) -> Pink (Rr) flowers).
    - These can be modeled for specific aesthetic traits or minor physiological traits by adjusting how the quantitativeEffectValues from the two alleles are combined (e.g., averaging for incomplete dominance, or a specific rule for co-dominance).
  + **Polygenic Interactions (Additive by Default, with Advanced Layers):**
    - **Additive Model (Primary):** As described in 5.2.1, the primary model for quantitative traits (yield, cannabinoids, terpenes) will be additive, where the effects of alleles across multiple genes sum up to determine genetic potential. This is a robust and computationally manageable approach for complex traits.
    - **Epistasis (Gene-by-Gene Interactions - Advanced Post-MVP):**
      * **Concept:** The effect of one gene is modified by one or more other genes (modifier genes).
      * **Implementation:** The AlleleDefinitionSO.interactionTriggers (List<AlleleInteractionRuleSO>) could point to rules that state: "IF Plant has Allele X at Gene A AND Allele Y at Gene B, THEN apply Z modifier to Trait T, OR enable/disable the effect of Allele Q at Gene C."
      * This requires a more complex GxE/TraitExpression calculator that checks for these epistatic conditions. It adds significant depth but also complexity to balancing and player understanding. Start with a few impactful epistatic interactions rather than many subtle ones.
    - **Pleiotropy (One Gene Affects Multiple Traits - Inherent):**
      * **Concept:** A single gene influences multiple, seemingly unrelated phenotypic traits.
      * **Implementation:** This is naturally handled if an AlleleDefinitionSO has quantitativeEffectValues for multiple different PhenotypicTraitSOs in its dictionary. For example, an allele for a key enzyme in a metabolic pathway might affect both a specific cannabinoid's production AND overall plant vigor.
      * This creates interesting trade-offs and correlations for breeders (e.g., breeding for high THC might inadvertently affect flowering time if a pleiotropic gene is involved).
  + **Mutation System (Full Vision):**
    - **Spontaneous Mutations:** A small, configurable background mutation rate (GlobalSettings.mutationRate). When an allele is passed from parent to offspring, there's a tiny chance it mutates.
    - **Types of Mutations:**
      * **Allele Switch:** Mutates to another *existing* AlleleDefinitionSO for that same GeneDefinitionSO (e.g., a "medium\_THCA" allele mutates to "high\_THCA" or "low\_THCA"). The probability of switching to a specific allele could be weighted by allele rarity or effect size.
      * **Novel Allele (Rare, Advanced):** A very rare chance of mutating into a "new" allele not previously common in the gene pool (could unlock a hidden AlleleDefinitionSO or dynamically generate a variant with slightly modified effect values). This is how new genetic diversity can be introduced.
      * **Loss-of-Function:** Mutates to a non-functional or null allele (e.g., an enzyme gene becomes inactive).
    - **Mutation Rate Modifiers:**
      * GeneDefinitionSO.mutationRateModifier can make certain genes or less prone to mutation.
      * Environmental stressors (e.g., simulated radiation, specific chemical mutagens if ever introduced as a game mechanic – very advanced) could temporarily increase mutation rates.
    - **Impact:** Mutations are a source of new genetic variation, essential for long-term breeding projects. They can be beneficial, neutral, or detrimental. This adds an element of chance and discovery.
  + **Linkage & Recombination (Very Advanced Future Consideration):**
    - **Concept:** Genes located close together on the same chromosome tend to be inherited together (linkage). Recombination (crossing-over) during meiosis can break these linkages.
    - **Implementation:** Would require defining gene positions on chromosome maps (GeneDefinitionSO.chromosomalLocus). The breeding logic would then need to simulate crossover events with probabilities based on genetic distance.
    - **Impact:** Makes it harder to combine desirable alleles from different linked genes if they are in repulsion phase, or easier if in coupling phase. Adds another layer of realism and challenge.
    - **Status:** This is likely beyond the scope of even the initial "full vision" due to its complexity but is a known factor in real-world breeding that could be considered for very late-game expert systems or expansions. For most of the game, independent assortment is a sufficient model.
  + **C# Implementation (Full Vision):**
    - BreedingManager.cs: Its BreedPlants() method will implement the core segregation, independent assortment, and mutation logic.
    - TraitExpressionCalculator.cs (or part of GeneticsManager): Will need to handle the additive model, and if implemented, check for dominance rules for specific qualitative traits, and process epistatic interaction rules from AlleleInteractionRuleSOs.
    - Data structures in PlantGenomeData, GeneDefinitionSO, and AlleleDefinitionSO will need to robustly store all necessary information for these complex interactions.
    - Custom editors for AlleleDefinitionSO and GeneDefinitionSO will be crucial for managing dominance relationships, effect values for multiple traits, and interaction triggers.
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the complex algorithms for mutation, epistatic checks, and the data structures for allele interactions. Generating boilerplate for the numerous SOs.

The full inheritance model aims to create a system where breeding is a challenging yet rewarding puzzle. Players will need to understand not just individual genes but how they interact with each other and the environment to truly master strain development.

**5.2.3. Foundational Strains (Initial 5 for MVP) & Expanding Library (Full Vision)**

* + **MVP Recap:** Five foundational strains, each with simple visual distinctions and underlying data structures for future complexity.
  + **Full Vision: A Vast and Diverse Genetic Pool:**
    - **Initial Foundational Strains (Expanded Detail):** The initial 5-10 "foundational" or "landrace-inspired" strains available at or near launch will be meticulously designed. Each PlantStrainSO for these will have:
      * A rich, pre-defined PlantGenomeData representing a plausible genetic makeup for a distinct archetype (e.g., a "classic Afghani Indica," a "tropical Sativa," a "high-CBD ruderalis hybrid").
      * Unique base values and optimal environmental parameters for all key PhenotypicTraitSOs, leading to distinct growth patterns, cannabinoid/terpene profiles (when GxE is fully active), and visual appearances.
      * Lore/description: A brief history or origin story for each foundational strain to add flavor.
    - **Introducing New Genetics Post-Launch (Key Content Strategy):**
      * **NPC Vendors & Specialists:** Unique NPC vendors who specialize in rare or exotic genetics will appear over time or as player reputation/progression increases. They might sell seeds or clones of new, unique foundational strains.
      * **Exploration & Discovery (Narrative Events - Doc1, Sec VII.D):**
        + "Genetic Expeditions": Late-game research projects or NPC-sponsored missions where the player funds or participates (abstractly) in expeditions to find "lost" landrace strains in remote regions. Success yields new, unique PlantStrainSOs with potentially novel alleles.
        + "Lore-Driven Discovery Quests": ADA or other NPCs might present quests or clues leading to the discovery of hidden genetic caches or ancient seed banks.
      * **Competitor Acquisitions (If Rival NPCs are Implemented):** Opportunities to acquire genetics from rival (NPC) cultivators through trade, corporate espionage (abstracted), or by purchasing their award-winning strains if they appear on a simulated market.
      * **Mutation as a Source:** As described in 5.2.2, spontaneous mutations can introduce new allelic variations into the player's existing gene pool.
      * **Community-Inspired (Potential Long-Term):** If modding or community content is ever supported, a framework for players to define and share their own PlantStrainSO data could be considered.
    - **Player's Personal Genetic Library:**
      * **Storage:** A robust in-game database (managed by PlayerGeneticsLibrary.cs and persisted via the save system - see Section 2.4.1) to store every unique PlantGenomeData the player creates through breeding or acquires.
      * **Organization:** Players can name their custom-bred strains, add detailed notes (e.g., breeding goals, observed phenotypes, GxE preferences), tag them with keywords, and filter/sort their library.
      * **Mother Plant Management:** Players can designate specific prized plants as "mother plants" for cloning, preserving their exact PlantGenomeData. The UI will clearly distinguish these.
      * **Seed Bank Management:** Detailed UI for storing, organizing, and viewing seeds, showing parental lineage, F-generation number, and predicted trait potentials.
    - **"Strain Uniqueness" Metric:** The game might internally track how genetically distinct a player-bred strain is from existing foundational strains or common archetypes. Highly unique and well-performing strains could gain "reputation" or higher market value.
    - **C# Implementation (Full Vision):**
      * PlantStrainSO becomes a highly detailed asset, with custom editors being essential for defining the complex genetic makeup and trait potentials of foundational strains.
      * PlayerGeneticsLibrary.cs: Manages the runtime collection of PlantGenomeData for all player-owned strains/seeds/clones. Handles saving/loading this extensive data.
      * Systems for procedural generation of new foundational strains (for NPC vendors or discovery events) based on predefined rules or archetypes, ensuring they have balanced and plausible genetic makeups.
    - **AI Tooling (Full Vision):**
      * Cursor AI: Generating C# for PlayerGeneticsLibrary.cs (data management, search/filter functions), and for the procedural generation logic for new NPC strains.
      * AI for Text Generation (e.g., GPT-3 via API if integrated, or offline use): Potentially assisting in generating draft lore descriptions or flavor text for new foundational strains, for human review and editing.

The expanding library of genetics is a core driver of long-term engagement, encouraging players to constantly explore, breed, and collect.

**5.2.4. Advanced Breeding Techniques (Post-MVP - Full Detail):** The full vision for breeding goes far beyond simple F1 crosses, incorporating established real-world breeding techniques that allow for strategic genetic refinement and stabilization. These are unlocked via the Skill Tree/Research system (Doc1, Sec VII.D).

* + **Backcrossing (BX):**
    - **Concept:** Repeatedly breeding a hybrid offspring back to one of its original parents (the "recurrent parent") over several generations (BX1, BX2, BX3, etc.). The goal is to isolate specific desirable traits from the non-recurrent parent while retaining the overall genetic background of the recurrent parent.
    - **Mechanics:**
      1. Player creates an F1 hybrid with a desired trait (e.g., unique aroma from Parent A) and good overall qualities from Parent B (recurrent).
      2. Player grows out F1s, selects an F1 individual expressing the desired trait, and crosses it back to Parent B. This creates BX1 offspring.
      3. Player grows out BX1s, selects individuals that still have the desired trait *and* most closely resemble Parent B, and crosses them back to Parent B again (creating BX2).
      4. This process is repeated. With each generation, the offspring become genetically closer to the recurrent parent, hopefully while retaining the targeted trait.
    - **Player Challenge:** Identifying offspring that carry the desired (potentially recessive) trait from the donor parent while also selecting for the recurrent parent's characteristics. Requires careful observation and potentially large population sizes.
    - **UI Support:** Pedigree tracking in the Genetics Lab UI becomes essential, clearly showing backcross generations.
  + **Inbred Lines (IBLs) & Strain Stabilization:**
    - **Concept:** Creating a highly homozygous (genetically uniform) strain by repeatedly self-pollinating a plant or breeding sibling plants from the same line over many generations (F2, F3, F4...F7+). The goal is to "lock in" desirable traits so the strain breeds true (offspring are very similar to parents).
    - **Mechanics:**
      1. Player starts with a hybrid (e.g., an F1 or a later generation) that exhibits a combination of traits they want to stabilize.
      2. **Selfing (S1):** If the plant can be induced to produce both male and female flowers (e.g., via a "Reversal Spray" consumable unlocked through research, simulating techniques like colloidal silver application), it can be self-pollinated to create S1 seeds. This rapidly increases homozygosity.
      3. **Sibling Breeding (Sib-Crossing):** Alternatively, player selects the best male and female siblings from a generation (e.g., F2s from an F1 x F1 cross) and breeds them together to create the next generation (F3).
      4. This process is repeated, with rigorous selection in each generation for plants that best express the desired combination of traits and show high uniformity.
    - **Player Challenge:** Managing inbreeding depression (reduced vigor, fertility, or expression of undesirable recessive traits as homozygosity increases). Requires careful selection and potentially outcrossing if depression becomes severe.
    - **Outcome:** A stabilized IBL will produce highly consistent offspring.
  + **Feminization Techniques (Creating Feminized Seeds):**
    - **Concept:** Producing seeds that are ~99% guaranteed to grow into female plants. This is highly desirable for growers focused on flower production, as male plants don't produce significant buds and can pollinate females, leading to seeded (less desirable) flowers.
    - **Mechanics (Abstracted In-Game):**
      1. Player selects a prized female plant they want to create feminized seeds from.
      2. Through research, they unlock a "Pollen Reversal Spray" or similar consumable/process.
      3. Applying this spray to a female plant (or a clone of it) during early flower induces it to produce male flowers (staminate flowers) that carry only female (X) chromosomes.
      4. Pollen from these reversed male flowers is then used to pollinate another (normal) female plant (ideally a clone of the same mother or a genetically similar female).
      5. The resulting seeds (S1 if selfed, or feminized F1s if crossed to a different female) will be feminized.
    - **Player Challenge:** The reversal process might have a success chance or temporarily stress the plant. Requires specific timing and resources.
  + **Pheno-Hunting (Meticulous Observation & Selection - Core Loop):**
    - **Concept:** The process of growing out a large number of plants from a new cross (especially F2 generations or later, where genetic variation is highest) to find rare, exceptional individuals ("keeper phenos") that exhibit the desired combination of traits. This is a cornerstone of breeding.
    - **Mechanics:**
      * Players will need significant grow space to pheno-hunt effectively.
      * Requires meticulous observation throughout the lifecycle: germination rates, seedling vigor, vegetative structure, flowering characteristics, resin production, aroma, stress resistance, and detailed post-harvest analysis (yield, lab reports for cannabinoids/terpenes).
      * The "Grower's Journal" (Doc1, Sec II.A) becomes an indispensable tool for tracking individual plants (e.g., by unique ID), their conditions, and observed traits.
      * Players select the very best individuals for further breeding or cloning as mother plants.
    - **UI Support:** Tools to easily compare stats and traits across many sibling plants. Ability to tag or star "keeper candidates."
  + **Genetic Marker Analysis (Late-Game Assist - Doc1, Sec VII.D):**
    - **Concept:** A late-game research unlock that allows players to take a small tissue sample from young seedlings and submit it to an in-game "lab" for analysis.
    - **Mechanics:** The lab report doesn't give definitive trait values but provides probabilistic "markers" indicating the likelihood of the seedling carrying desirable (or undesirable) alleles for key traits that are hard to observe visually in young plants (e.g., high THC potential, specific terpene synthase genes, disease resistance genes).
    - **Player Benefit:** Helps in culling unpromising seedlings early during large pheno-hunts, saving space and resources. It's suggestive, not definitive – GxE still plays a role, and some traits only express later.
    - **Cost:** This service would be expensive in-game (currency, rare resources, time for results).
  + **C# Implementation (Full Vision):**
    - BreedingManager.cs will need methods for BackCross(), SelfPollinate(), CreateFeminizedSeeds(). These will involve specific logic for selecting parental PlantGenomeData and generating offspring genomes.
    - PlantInstance.cs or PlantGenomeData.cs will need to track generation numbers (F1, F2, BX1, S1, etc.).
    - The "Reversal Spray" will be an ConsumableItemSO with an associated effect script that temporarily changes a female plant's state to produce male-type pollen.
    - GeneticMarkerAnalysisLab.cs: A system that takes a PlantGenomeData (from a seedling) as input, simulates a probabilistic analysis based on its alleles for marker-linked traits, and generates a report. The probabilities would be defined in GeneDefinitionSO or AlleleDefinitionSO (e.g., "Allele X is 80% correlated with Marker Y for High THC").
    - UI for pedigree tracking, detailed plant comparison for pheno-hunting, and displaying genetic marker reports.
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the C# logic for the new breeding methods, the state changes for pollen reversal, and the probabilistic calculations for genetic marker analysis.

These advanced breeding techniques transform genetics from simple F1 creation into a deep, strategic, multi-generational endeavor, allowing players to truly sculpt their ideal cannabis strains.

**\*\*5.2.5. The "AI Research Lab"**: Predictive Breeding (Post-MVP - Full Detail)\*\*  
The "AI Research Lab" is a significant late-game feature, unlocked via substantial progression in the Science and Genetics skill trees/research system. It represents a paradigm shift in the player's breeding capabilities, moving from purely experimental observation to data-driven, AI-assisted predictive breeding. (Doc1, Sec VII.D; Doc2, Sec II.B).  
  
\* \*\*Concept & Player Experience:\*\*  
 \* The AI Research Lab is an in-game facility upgrade or a dedicated room/module the player constructs and equips. It's presented to the player as a sophisticated computational genetics laboratory.  
 \* Players can submit the `PlantGenomeData` of two potential parent plants to the AI Lab.  
 \* The Lab then runs complex simulations (the "AI") to predict the \*probabilistic distribution\* of key polygenic traits in their F1 offspring.  
 \* \*\*Output/UI:\*\* Instead of a single definitive outcome, the AI Lab provides:  
 \* \*\*Predicted Trait Ranges:\*\* For traits like THC potential, yield, flowering time, it might show a bell curve or a range (e.g., "Predicted THC Potential: 18-26%, with highest probability around 22%").  
 \* \*\*Probability of Specific Allele Combinations:\*\* Likelihood of offspring inheriting desired (or undesired) homozygous or heterozygous allele pairs for key genes.  
 \* \*\*Potential for Novel Trait Expression (if epistasis is modeled):\*\* May highlight potential for unexpected trait expressions due to specific gene interactions.  
 \* \*\*Warnings:\*\* Flag potential negative outcomes like increased susceptibility to a disease if both parents carry recessive susceptibility alleles, or risk of inbreeding depression if parents are closely related.  
 \* \*\*Suggested Parental Combinations (Advanced):\*\* Players might be able to set a "target profile" (e.g., "High THC, short flowering time, good mold resistance"), and the AI Lab could analyze the player's entire genetic library to suggest the top 3-5 parental pairings most likely to achieve offspring close to that target.  
 \* \*\*Resource Cost:\*\* Using the AI Research Lab is not free. It will consume significant in-game resources:  
 \* \*\*Currency:\*\* For "computation time" or "lab fees."  
 \* \*\*Rare Materials/Data Chips:\*\* Consumables required to run complex simulations.  
 \* \*\*In-Game Time:\*\* Simulations take time to run (e.g., several in-game days or weeks), requiring players to plan their research.  
 \* \*\*Augmentation, Not Replacement:\*\* The AI Lab \*augments\* player skill and observation; it doesn't replace it. Predictions are probabilistic, not guarantees. Pheno-hunting is still essential to find the actual exceptional individuals within the predicted range. The AI helps narrow down choices and manage the vast combinatorial complexity of polygenic breeding.  
  
\* \*\*Underlying "AI" Mechanics (How it "Works" In-Game):\*\*  
 \* \*\*Initial Implementation (Simplified Algorithms - Doc1, Sec VII.D):\*\*  
 \* The "AI" could initially be sophisticated algorithms based on the game's own genetic inheritance rules (segregation, independent assortment, additive allele effects, mutation rates).  
 \* It would essentially run many thousands of simulated breeding events between the two parent genomes in an accelerated, non-visual way, tabulating the resulting trait distributions. This is computationally intensive but deterministic based on the game's rules.  
 \* \*\*Advanced Implementation (Unity Sentis - Doc2, Sec II.B; Doc1, Sec IV.C):\*\*  
 \* For more nuanced or "smarter" predictions, especially if complex epistatic interactions or GxE pre-dispositions are heavily modeled:  
 1. \*\*Offline Model Training:\*\* Develop and train machine learning models (e.g., Bayesian networks, regression models, or even small neural networks) \*outside\* the game. The training data would be generated by running millions of simulated breeding crosses and GxE expressions using Project Chimera's core genetic and environmental simulation engines. The model would learn to predict offspring trait distributions from parental genotypes.  
 2. \*\*Model Conversion & Sentis Integration:\*\* Convert the trained ML model to ONNX format and integrate it into Unity using Sentis.  
 3. \*\*Runtime Inference:\*\* When the player submits parents to the AI Lab, their `PlantGenomeData` is fed as input to the Sentis-powered model, which then outputs the probabilistic trait predictions.  
 \* This approach allows for more complex pattern recognition and potentially more "intelligent" suggestions than purely rule-based simulations, but requires significant ML expertise and infrastructure for model training.  
  
\* \*\*Progression & Upgrades:\*\*  
 \* The AI Research Lab itself could be upgradeable. Initial version might offer predictions for fewer traits or with wider probability ranges.  
 \* Research projects or skill tree unlocks could improve the Lab's accuracy, speed, number of traits analyzed, or unlock advanced features like the "suggest parental combinations" tool.  
  
\* \*\*C# Implementation (Full Vision):\*\*  
 \* `AIResearchLabController.cs`: Manages the AI Lab UI, job submission, resource consumption, simulation timing, and display of results.  
 \* `PredictiveBreedingEngine.cs`: Contains the core logic for simulating breeding outcomes.  
 \* If rule-based: Implements the high-speed Monte Carlo simulation of genetic inheritance.  
 \* If Sentis-based: Handles preparing input tensors for the Sentis model, running inference, and parsing output tensors.  
 \* `BreedingPredictionReport.cs`: Data structure to hold the results of a prediction (trait distributions, probabilities, warnings).  
 \* UI scripts for displaying complex probabilistic data in an understandable way (charts, graphs, confidence intervals).  
\* \*\*AI Tooling (Full Vision):\*\*  
 \* Cursor AI: Assisting with the C# logic for the `AIResearchLabController`, the rule-based `PredictiveBreedingEngine`, and the UI data presentation.  
 \* Unity Sentis: The runtime inference engine if ML models are used.  
 \* Python/ML Frameworks (TensorFlow, PyTorch): For offline training of predictive ML models.  
  
The AI Research Lab is a key late-game system that elevates breeding from pure trial-and-error to a more strategic, data-informed science, reinforcing Project Chimera's theme of "scientific breakthroughs" and "data-driven decision making."  
  
**\*\*5.2.6. Trait Library:** Player-Populated Genetic Discoveries (Full Vision)\*\*  
  
\* \*\*MVP Recap:\*\* Simple UI list where discovered visual MVP traits are automatically logged.  
\* \*\*Full Vision: Comprehensive, Interactive Genetic Database:\*\*  
 \* \*\*Automatic Logging of All Traits:\*\* Every time a plant expresses a phenotypic trait (visual, cannabinoid, terpene, growth characteristic, resistance, etc.) for the first time, or a new \*level\* of a quantitative trait is achieved (e.g., first plant to hit 25% THC), that trait and its expression level are logged in the Trait Library.  
 \* \*\*Detailed Trait Information:\*\* For each entry, the Library displays:  
 \* `PhenotypicTraitSO` data: Name, description, unit of measurement.  
 \* Range of expression observed by the player for this trait across all their plants.  
 \* The specific plant(s) that first exhibited this trait or its peak expression.  
 \* Links to relevant `GeneDefinitionSO`s and `AlleleDefinitionSO`s known to influence this trait (unlocked via research or analysis in the AI Research Lab).  
 \* Player notes section for each trait.  
 \* \*\*Filtering & Sorting:\*\* Players can filter the Trait Library by trait type (e.g., Cannabinoids, Terpenes, Growth), search by name, and sort by various criteria.  
 \* \*\*"Discovery" Mechanic:\*\*  
 \* Some alleles, especially rare recessive ones or those leading to novel epistatic interactions, might be "hidden" initially. When a plant expresses a trait due to such a hidden genetic factor, it's a "Discovery" event, prominently logged in the Library.  
 \* This provides a "gotta catch 'em all" meta-game for genetic explorers.  
 \* \*\*Integration with Breeding UI & AI Research Lab:\*\*  
 \* When selecting parents, the breeding UI can cross-reference with the Trait Library to show which known traits each parent possesses or is likely to pass on.  
 \* The AI Research Lab can use the Trait Library data to inform its predictions (e.g., if the player has discovered a rare allele, the Lab can incorporate its effect into simulations).  
 \* \*\*Visual Representation:\*\* The Trait Library UI will be a rich, database-like interface, potentially with visual icons or representations for different traits.  
 \* \*\*C# Implementation (Full Vision):\*\*  
 \* `TraitLibraryManager.cs`: Manages the persistent collection of discovered `DiscoveredTraitData` (which would link to `PhenotypicTraitSO` and store player-specific observation details like max value seen, discovery plant ID, etc.).  
 \* `DiscoveredTraitData.cs`: Runtime and saveable data structure.  
 \* Robust UI scripts for displaying, filtering, and sorting the library.  
 \* Event listeners that monitor plant trait expression (e.g., `OnPlantHarvestedWithTraitReportEvent`) and update the Trait Library.  
 \* \*\*AI Tooling (Full Vision):\*\*  
 \* Cursor AI: Logic for `TraitLibraryManager`, data structures, UI backend.  
  
The Trait Library evolves from a simple MVP list into a comprehensive, personalized encyclopedia of the player's genetic journey, documenting their discoveries and serving as a vital research tool for advanced breeding.  
  
**\*\*5.2.7. Tissue Culture & Micropropagation (Advanced Post-MVP - Full Detail)\*\***  
This is a late-game, high-tech cloning method offering significant advantages over traditional cuttings, unlocked via advanced Science/Genetics research.  
  
\* \*\*Concept:\*\* In vitro cultivation of plant tissues or cells to produce new plants (clones) under sterile laboratory conditions. Offers rapid multiplication of elite genetics and a way to "clean" plants of certain systemic diseases.  
\* \*\*Mechanics:\*\*  
 1. \*\*Dedicated Lab Equipment:\*\* Requires a "Tissue Culture Lab" facility module or a set of specialized equipment:  
 \* `LaminarFlowHoodSO`: Provides a sterile workspace.  
 \* `AutoclaveSO`: For sterilizing media and tools.  
 \* `GrowthMediumPreparationStationSO`: For mixing agar, nutrients, hormones.  
 \* `IncubationChamberSO` / `GrowthShelvesSO`: With controlled light and temperature for growing cultures.  
 \* Consumables: `AgarSO`, `PlantHormoneSO` (auxins, cytokinins), `SterileContainerSO` (petri dishes, culture vessels).  
 2. \*\*Taking Explants:\*\* Player selects a healthy, prized mother plant. An action allows them to take small tissue samples ("explants") from it (e.g., shoot tips, nodes). This might be a minigame requiring precision or have a success chance.  
 3. \*\*Culture Initiation:\*\* Explants are placed onto sterile growth medium in containers within the Laminar Flow Hood. The composition of the medium (especially hormone balance) is critical and can be customized by the player (based on research unlocks) to promote callus formation, shoot proliferation, or rooting.  
 4. \*\*Multiplication Stage:\*\* Callus tissue or shoot cultures are repeatedly subdivided and transferred to fresh medium to rapidly multiply the number of potential plantlets.  
 5. \*\*Rooting & Acclimatization:\*\* Shoots are transferred to a rooting medium. Once rooted, the delicate plantlets must be carefully acclimatized from the high-humidity, sterile lab environment to normal greenhouse conditions. This is a sensitive stage with a risk of losses.  
\* \*\*Benefits:\*\*  
 \* \*\*Rapid Multiplication:\*\* Produce hundreds or thousands of clones from a single mother plant much faster than traditional cuttings.  
 \* \*\*Genetic Preservation:\*\* Ideal for long-term storage of elite genetics in vitro.  
 \* \*\*Disease Cleaning (Abstracted):\*\* Meristem culture (using the very tip of a shoot) can sometimes eliminate certain viruses or systemic pathogens from a plant line. This could be a game mechanic: tissue-cultured clones might have a "Clean" status or higher base health/resistance.  
\* \*\*Challenges & Costs:\*\*  
 \* High initial investment in lab equipment and research.  
 \* Requires meticulous sterile technique (contamination is a risk, leading to culture loss – simulated via probability or minigame).  
 \* Consumable costs for media, hormones, sterile supplies.  
 \* Time-consuming process with multiple stages.  
\* \*\*C# Implementation (Full Vision):\*\*  
 \* `TissueCultureLabController.cs`: Manages the overall process, equipment states, and culture progression.  
 \* `PlantCulture.cs`: Runtime class representing an active tissue culture, tracking its stage (callus, shoots, rooting), health, contamination risk, and multiplication factor.  
 \* State machine for culture progression.  
 \* UI for managing cultures, preparing media recipes, and monitoring incubation chambers.  
 \* Minigames or skill-check interactions for sterile procedures or explant taking.  
\* \*\*AI Tooling (Full Vision):\*\*  
 \* Cursor AI: Logic for `TissueCultureLabController`, `PlantCulture` state management, media recipe calculations.  
 \* AI for Concept Art: Visualizing the high-tech tissue culture lab equipment.  
  
Tissue culture adds a layer of high-end scientific gameplay, appealing to players who want to delve into advanced propagation methods and large-scale genetic preservation/multiplication.  
  
**\*\*5.2.8. Genetic Acquisition:** Landraces, Expeditions, NPC Contacts (Full Vision)\*\*  
Expanding the player's genetic pool with new, unique foundational material is crucial for long-term breeding diversity. The MVP starts with 5 strains; the full game needs mechanisms to acquire more. (Doc1, Sec VII.D).  
  
\* \*\*Landrace Strains:\*\*  
 \* \*\*Concept:\*\* Ancient, geographically isolated cannabis varieties that are genetically distinct and often homozygous. They represent a valuable source of unique traits and resilience.  
 \* \*\*Acquisition Methods:\*\*  
 \* \*\*NPC Specialist Vendors:\*\* Rare NPCs who occasionally offer seeds or clones of landrace strains (e.g., "Afghani Kush," "Colombian Gold," "Thai Stick") for a very high price or in exchange for rare player-bred strains/products.  
 \* \*\*Genetic Expeditions (Late-Game Research/Narrative):\*\*  
 \* Player funds and directs (abstractly) expeditions to remote, exotic locations known for specific landraces.  
 \* Involves research, resource investment (currency, specialized personnel/equipment as conceptual items), and time.  
 \* Success is probabilistic, influenced by research level and investment. May yield multiple attempts.  
 \* Successful expeditions return with a few seeds/clones of a new landrace `PlantStrainSO`, along with an "Acquisition Report" or "Origin Dossier" detailing its purported origin, characteristics, and lore.  
 \* \*\*Lore-Driven Discovery Quests:\*\* ADA or other NPCs might provide clues or quests (e.g., "analyze this ancient text," "investigate anomaly in Sector Gamma's old seed vault") that lead to the discovery of a forgotten landrace.  
\* \*\*Elite Clones / Cultivars from other NPCs:\*\*  
 \* \*\*High-Tier NPC Contacts:\*\* As player reputation grows, they might gain access to elite breeders or cultivators (NPCs) who are willing to trade or sell clones of their prized, stabilized cultivars. These would be unique, high-performance `PlantStrainSO`s.  
 \* \*\*"Cannabis Cup" Style Events (Post-MVP Economy):\*\* If in-game competitions are introduced, winning entries from NPCs might become available for acquisition (at a high price or as a reward).  
\* \*\*"Acquisition Report" / "Origin Dossier":\*\*  
 \* When new foundational genetics are acquired, they come with a lore item providing:  
 \* Strain Name & Origin (fictional or inspired by real-world).  
 \* Traditional Uses / Cultural Significance (lore).  
 \* Key Observed Characteristics in its native environment.  
 \* Potential unique alleles or traits it might carry.  
 \* This adds flavor and guides the player on how the new genetics might be useful.  
\* \*\*C# Implementation (Full Vision):\*\*  
 \* `GeneticAcquisitionManager.cs`: Handles logic for NPC vendor inventories, expedition system (success chances, reward generation), and quest-driven discoveries.  
 \* `Expedition.cs`: Runtime class for managing an ongoing expedition (progress, resource needs, outcome).  
 \* `LoreItemSO.cs` for Acquisition Reports/Dossiers.  
 \* New `PlantStrainSO` assets for each acquirable landrace or elite cultivar, with pre-defined (but potentially partially hidden initially) genetic data.  
\* \*\*AI Tooling (Full Vision):\*\*  
 \* Cursor AI: Logic for `GeneticAcquisitionManager`, `Expedition` state.  
 \* AI for Text Generation: Assisting in drafting lore for Acquisition Reports, NPC dialogue related to genetic trading.  
 \* AI for Image Generation: Creating visuals for expedition locations, NPC specialist vendor portraits, or icons for new strains.  
  
These acquisition mechanics ensure a continuous influx of new genetic material into the game world, fueling the player's breeding programs and providing long-term collection and discovery goals.

### 5.3. Environmental Control Systems: Mastering the Microclimate

The ability to precisely monitor, understand, and manipulate the cultivation environment is absolutely central to successful GxE interaction and, therefore, to mastering Project Chimera. While the MVP introduces very basic manual controls and standalone sensors, the full vision encompasses sophisticated, interconnected utility networks, detailed microclimate simulation, and advanced automation capabilities, transforming environmental management into a deep engineering and strategic challenge.

* **MVP Recap (as per Part 4, Sec 4.3.3):**
  + **Manual Adjustments:** On/off toggles for basic lights and fans. Rudimentary heaters/coolers (if in Residential House MVP).
  + **Basic Standalone Sensors:** Manually checked digital thermometers/hygrometers.
  + **Rudimentary Automation:** Simple light timers. Very basic on/off thermostats/humidistats integrated into specific equipment.
  + **Focus:** "Burden of Consistency" to teach fundamentals and motivate progression towards automation.
  + **Exclusions:** No networked sensors, central dashboards, PLCs, advanced HVAC, detailed airflow, CO2 control.
* **Full Vision: Integrated, Automated, and Precisely Controlled Environments:**

**5.3.1. Advanced Utility Networks:** Plumbing, Electrical, HVAC (Full Detail)  
The full vision moves far beyond the MVP's abstracted utilities, requiring players to design, build, and manage complex, interconnected 3D utility networks for power, water/nutrients, and air handling, particularly in the Warehouse and future advanced facilities. This is a core "Satisfactory-like" infrastructure management pillar (Doc1, Sec I.A).

* + **Detailed 3D Routing & Components:**
    - **Plumbing System (Water & Nutrient Solutions):**
      * **Components:** PipeSO (various diameters/materials affecting flow rate, pressure tolerance, cost), ValveSO (manual on/off, automated/solenoid), PumpSO (various capacities, power draws), ReservoirSO (various sizes, insulated/chilled options), FilterSO (water/nutrient solution filters), FittingSO (elbows, tees, connectors).
      * **Player Interaction:** Players manually route pipes in 3D space, connecting reservoirs to pumps, pumps to filters, and then to distribution manifolds or directly to hydroponic systems/irrigation emitters (e.g., drip lines, spray nozzles for soil/coco).
      * **Simulation:**
        + **Flow Rate & Pressure (Abstracted):** While not a full fluid dynamics simulation, the system will model flow rates based on pipe diameter, pump capacity, and total demand. Abstracted "pressure" will determine if water can reach all endpoints, especially in multi-level facilities. Gravity might play a simplified role.
        + **Solution Integrity:** Different nutrient solutions should not mix in pipes unless intended (e.g., via a mixing valve). Contamination (if a disease enters a reservoir) can spread through connected plumbing.
    - **Electrical System:**
      * **Components:** WireSO (different gauges affecting amperage capacity, cost), JunctionBoxSO, CircuitBreakerSO (trips if overloaded), PowerGeneratorSO (solar panels, gas generators, grid connection point – with varying outputs and costs), BatteryBankSO (for backup power), TransformerSO (if different voltage tiers are simulated – advanced).
      * **Player Interaction:** Players route wires from power sources to circuit breakers, then to junction boxes, and finally to individual pieces of equipment.
      * **Simulation:**
        + **Power Load & Capacity:** Each piece of equipment has a power draw (EquipmentDataSO.powerConsumption). The total load on a circuit must not exceed the breaker's rating or the wire's capacity. Overloads trip breakers, shutting down equipment on that circuit.
        + **Voltage Tiers (Optional Advanced):** Could introduce different voltage requirements for light vs. heavy machinery, requiring transformers.
        + **Power Grid Management:** Balancing power generation with consumption. Managing battery charge/discharge cycles. Potential for brownouts or equipment damage if power is unstable (very advanced).
    - **HVAC (Heating, Ventilation, Air Conditioning) System:**
      * **Components:** DuctSO (various sizes/insulation affecting airflow efficiency), InlineFanSO (boosts airflow in ducts), IntakeVentSO (brings outside air in, potentially with filters), ExhaustVentSO (expels stale air), AirConditioningUnitSO (central or split units), HeaterUnitSO (central or localized), HumidifierSO/DehumidifierSO (central/in-duct or standalone), CO2GeneratorSO/CO2TankSO & CO2DistributionNozzleSO.
      * **Player Interaction:** Players design and route ductwork connecting intake/exhaust vents, AC/heater units, and distribution vents/nozzles within grow rooms.
      * **Simulation:**
        + **Airflow Rates (Abstracted):** Duct size, fan power, and ductwork complexity (bends, length) influence airflow volume (CFM/CMH) delivered to/from rooms.
        + **Air Exchange Rate:** Affects CO2 replenishment, humidity removal, and temperature homogenization within a room.
        + **Filtered Air:** Intake filters can reduce incoming pest/spore load (abstracted).
        + **CO2 Distribution:** CO2 from generators/tanks is distributed via ducting to maintain target PPM levels in grow rooms.
  + **Logical Source-to-Endpoint Connections & Visual Feedback:**
    - For any utility system to function, a complete, logical path must exist from a source (e.g., reservoir, generator, CO2 tank) to an endpoint (e.g., plant irrigation, light, CO2 nozzle).
    - The UI will provide clear visual feedback on successful connections (e.g., pipes/wires highlighting, flow animations as in Doc1, Sec VI.B) and errors (e.g., unconnected segments, overloaded circuits, insufficient pressure).
  + **C# Implementation (Full Vision for Utility Networks):**
    - **Graph-Based Representation:** Each utility network (plumbing, electrical, HVAC) can be modeled as a graph where equipment and junctions are nodes, and pipes/wires/ducts are edges with properties (capacity, resistance, etc.).
    - UtilityNetworkManager.cs (potentially one per type: PlumbingNetworkManager, ElectricalNetworkManager, HVACNetworkManager):
      * Manages the graph structure.
      * Performs flow/load calculations (e.g., using algorithms like Ford-Fulkerson for max flow in plumbing, or simple summation for electrical load against circuit capacity).
      * Handles updates when players build/modify networks.
      * Provides data to the EnvironmentController (e.g., how much cooling an AC unit connected to a specific room's ductwork can provide).
    - PlaceableUtilityComponent.cs (base class for pipes, wires, ducts, valves, junctions): Handles placement logic, connection validation, and updates its status in the network graph.
    - UtilityNode.cs and UtilityEdge.cs: Data structures for the graph.
  + **AI Tooling (Full Vision for Utility Networks):**
    - Cursor AI: Assisting with the graph data structures, pathfinding/flow algorithms within the network managers, and the C# logic for PlaceableUtilityComponent connection validation.
    - AI for Concept Art: Visualizing complex utility layouts or high-tech component designs.

This detailed utility network system transforms facility design into a significant engineering puzzle, directly impacting the player's ability to scale operations and maintain optimal environmental conditions.

**5.3.2. The "X-Ray" Utility View Toggle:** Visualizing Hidden Infrastructure (Full Detail)Managing the complex, potentially multi-layered utility networks described above would be nearly impossible without a dedicated visualization tool. The "X-Ray" Utility View is critical for this. (Doc1, Sec VI.B; Doc2, Sec IV.B).

* + **MVP Recap:** Not available in the Residential House. Becomes essential upon unlocking the Warehouse for custom construction with potentially hidden utilities.
  + **Full Vision Functionality:**
    - **Toggleable View Mode:** Player can activate this view via a UI button or hotkey.
    - **Structural Translucency:** When active, all primary structural elements of the facility (walls, floors, ceilings, potentially large static equipment casings) render as "ghostly translucent" or wireframe. This allows players to see *through* them.
    - **Utility Network Highlighting:**
      * All placed utility components (pipes, wires, ducts) within the translucent structures become clearly visible and opaque.
      * **Color-Coding:** Different utility types are distinctly color-coded for immediate identification:
        + E.g., Water pipes: Blue; Nutrient solution pipes (if separate): Green; Electrical wires: Yellow/Orange; HVAC Ducts (Supply): Light Blue; HVAC Ducts (Return): Light Red; CO2 lines: White.
      * **Status Indication (Advanced):**
        + Active/Flowing: Components might have subtle emissive glows, animated pulses, or brighter colors if currently transporting resources/power.
        + Inactive/Problematic: Components that are unpowered, unconnected, blocked, or part of an overloaded circuit might be colored differently (e.g., greyed out, flashing red).
      * **Selected Component Highlighting:** When the player hovers over or selects a utility component in X-Ray view, it (and potentially its directly connected segment) highlights more intensely.
    - **Interaction in X-Ray View:** Players can still select, place, modify, and delete utility components while in X-Ray view, making it an active construction and troubleshooting mode, not just a passive visualization.
    - **Information Overlay (On Hover/Select):** When a utility component is selected in X-Ray view, a UI tooltip or panel displays relevant information:
      * Pipe: Diameter, material, current flow rate/pressure, solution type.
      * Wire: Gauge, current load, max capacity, circuit ID.
      * Duct: Size, airflow rate, air temperature/humidity/CO2 within.
    - **Filtering (Advanced):** Option to toggle visibility of specific utility layers (e.g., "Show only Electrical," "Hide Plumbing").
  + **Importance & Player Experience:**
    - **Essential for Complex Construction:** Indispensable for planning and routing utilities in multi-story facilities or dense equipment layouts where networks run within walls or above ceilings.
    - **Troubleshooting:** Makes it vastly easier to identify breaks in a network, bottlenecks, overloaded circuits, or incorrectly connected components.
    - **Optimization:** Allows players to analyze their utility layouts for efficiency (e.g., minimizing pipe/wire lengths, ensuring balanced loads).
    - The X-Ray view transforms abstract utility management into a tangible, visual engineering task, strongly aligning with the "Detailed Infrastructure Management" pillar.
  + **C# Implementation (Full Vision):**
    - XRayViewController.cs: Manages the toggling of the view mode.
    - When activated, it iterates through all structural GameObjects (tagged appropriately) and applies a "translucent" or "wireframe" shader/material. It iterates through all utility GameObjects and ensures they use their normal opaque, color-coded, and potentially status-highlighted materials.
    - Requires a robust system for tagging or layering GameObjects as "Structural" vs. "Utility\_Electrical," "Utility\_Plumbing," etc.
    - Shader development for the translucent effect and potentially for the utility highlighting/flow animations. MaterialPropertyBlocks will be key for dynamically changing utility component appearances (color, emission) based on status without creating many material instances.
    - Integration with the selection and information display systems to show contextual data for utility components in this mode.
  + **AI Tooling (Full Vision):**
    - Not directly for generating the X-Ray view itself, but AI-generated equipment assets must be designed with clear visual distinction that works well with color-coding in this view.

The X-Ray Utility View is not just a visual gimmick; it's a fundamental gameplay tool for managing the intricate circulatory systems of the player's advanced cultivation facilities.

**5.3.3. Microclimate Modeling:** Localized Environmental Variations (Full Detail)To elevate the environmental simulation beyond simple room-wide averages, the full vision includes modeling localized microclimates within controlled spaces. This means conditions can vary within a single grow room based on equipment placement, airflow, and plant density. (Doc2, Sec IV.C).

* + **MVP Recap:** Environmental parameters are largely room-wide averages. Basic fans provide a generic localized "improved airflow" buff.
  + **Full Vision Functionality:**
    - **Grid-Based Environmental Simulation (within Rooms):**
      * Each enclosed grow room (or large, distinct zone) maintains an internal 3D grid of "environment cells" or voxels (as described in Doc3, Sec 2.3.1).
      * Each cell tracks its own local values for: Temperature, Humidity, CO2 level, Light Intensity (PAR/PPFD), and potentially an "Airflow Velocity/Quality" metric.
    - **Equipment Influence Propagation:**
      * **Sources & Sinks:** Environmental control equipment (heaters, coolers, humidifiers, dehumidifiers, CO2 emitters, lights, fans) act as sources or sinks for specific parameters.
      * **Radius/Cone of Influence:** Each piece of equipment projects an "influence" into the surrounding environment cells. The strength of this influence typically diminishes with distance from the equipment.
        + *Example:* A small electric heater primarily heats the cells closest to it, with heat gradually propagating outwards. A fan creates a cone of increased airflow. A grow light has a defined PAR footprint.
      * **Obstructions:** Dense plant canopies, walls, or other large equipment can obstruct the propagation of influence (e.g., a dense canopy can create a pocket of higher humidity and lower airflow beneath it, or block light from reaching lower leaves). This requires simplified raycasting or voxel-based occlusion checks from influence sources.
    - **Environmental Parameter Propagation & Diffusion:**
      * **Temperature:** Heat propagates between adjacent cells via simplified conduction and convection (e.g., warmer cells have a tendency to transfer some heat to cooler neighbors, with a slight upward bias for convection). Heat loss/gain through walls (based on WallSO.insulationRValue) affects cells adjacent to exterior walls.
      * **Humidity:** Water vapor diffuses from cells of higher RH to lower RH. Plant transpiration adds humidity to cells occupied by plants. Dehumidifiers remove it locally.
      * **CO2:** Diffuses similarly to humidity. Plants consume CO2 in illuminated cells (photosynthesis). CO2 emitters enrich local cells. Air exchange with outside (via vents or leaks) brings in fresh air with ambient CO2.
      * **Airflow:** Fans generate "airflow vectors" or increase an "airflow quality" metric in cells within their cone of influence. This airflow helps homogenize temperature/humidity/CO2, reduces boundary layers around leaves (improving transpiration/CO2 uptake), and can help dissipate pockets of stale air.
    - **Plant Canopy Effects:** Dense plant canopies create their own microclimate:
      * Reduced light penetration to lower levels.
      * Increased local humidity due to transpiration.
      * Reduced airflow within the canopy.
      * These effects mean that sensors placed above the canopy might not reflect conditions *within* it, encouraging players to place sensors strategically.
    - **Sensor Placement Matters:** The readings from player-placed sensors (StandaloneSensor.cs in MVP, networked AdvancedSensorSOs later) will reflect the specific conditions of the environment cell they occupy. This makes strategic sensor placement crucial for accurate monitoring and control.
    - **Computational Cost Management:**
      * The environment grid update frequency needs to be balanced for performance. Not every cell needs to be updated every frame. Updates can be staggered or occur at lower frequencies (e.g., every few seconds or when significant equipment state changes occur).
      * Simplified propagation algorithms (e.g., cellular automata-like rules, weighted averaging with neighbors) rather than full computational fluid dynamics (CFD).
  + **Player Experience & Strategic Implications:**
    - Players must think more three-dimensionally about their grow room layouts. Equipment placement directly impacts local conditions around plants.
    - "Hot spots" or "cold spots," areas of high/low humidity, or pockets of stagnant air can develop if airflow is poor or equipment is badly placed.
    - Optimizing airflow using fans and ducting (from the HVAC system) becomes a key strategy for creating uniform conditions and maximizing plant potential across the entire grow space.
    - Encourages experimentation with different equipment layouts and sensor placements to achieve ideal microclimates for specific strains or growth stages.
  + **C# Implementation (Full Vision):**
    - EnvironmentCell.cs (Struct or Class): Stores local temp, humidity, CO2, light, airflow values.
    - RoomEnvironmentController.cs:
      * Manages the 3D grid of EnvironmentCells for its room.
      * Periodically runs the propagation/diffusion simulation update loop.
      * Receives influence data from active EnvironmentalControlEquipment.cs instances within the room.
      * Provides cell-specific data to sensors or plants querying their local conditions.
    - EnvironmentalControlEquipment.cs (and derived classes like Heater.cs, Fan.cs, GrowLight.cs):
      * Each will have methods to calculate its "influence map" or project its effect onto the RoomEnvironmentController's grid (e.g., ApplyHeatInfluence(EnvironmentCell[,] grid), ApplyAirflowInfluence(EnvironmentCell[,] grid)).
      * Grow lights will need a more complex CalculateLightDistribution(EnvironmentCell[,] grid) method, potentially involving simplified raycasting or pre-calculated footprints.
    - PlantInstance.cs: Queries its RoomEnvironmentController for the conditions in the cell(s) it occupies to feed into its GxE calculations. Its own transpiration and physical presence (occlusion) will also feedback as an influence on its local cells.
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the grid management logic in RoomEnvironmentController, the propagation algorithms, and the influence calculation methods for various equipment types.
    - AI for Visualization (Developer Tool): Potentially, an AI tool could be used to analyze simulation data from many microclimate runs and help visualize complex airflow patterns or heat distribution in a way that aids developer understanding and balancing, though this is highly speculative.

Microclimate modeling adds a significant layer of depth and realism to environmental control, rewarding players who master the nuances of airflow, equipment placement, and spatial environmental dynamics.

**5.3.4. Advanced Automation:** Sensors, Controllers, PLCs (Post-MVP - Full Detail)The "Earned Automation" philosophy (Doc1, Sec VII.C) culminates in a suite of advanced sensors and programmable controllers, allowing players to design sophisticated, responsive environmental management systems for large-scale or precision-critical operations.

* + **MVP Recap:** Basic light timers, very simple integrated on/off thermostats/humidistats. Manual sensor checking.
  + **Full Vision Functionality:**
    - **Tiered Sensor Systems (SensorSO - Basic, Intermediate, Advanced - Doc1, Sec VII.C):**
      * **Basic Sensors (MVP carry-over):** Standalone, manually checked (thermometer, hygrometer, handheld EC/pH).
      * **Intermediate Networked Sensors:**
        + Can be connected to the facility's data network (a conceptual utility layer, perhaps sharing wiring with low-voltage electrical or its own data cables).
        + Transmit readings to a central EnvironmentalMonitoringDashboardUI.
        + Types: Networked Temperature, Humidity, VPD, CO2, basic Soil Moisture Probes (for pots), basic inline Water EC/pH Sensors (for reservoirs).
        + Accuracy: Better than basic, but still some margin of error.
      * **Advanced Precision Sensors:**
        + Highly accurate, faster response times, more specialized.
        + Types: Leaf Surface Temperature Probes, PAR/PPFD Quantum Sensors (for precise light measurement at canopy level), advanced multi-parameter Soil/Substrate Probes (moisture, EC, temp, pH), high-accuracy inline Water Sensors, Air Particle/Spore Traps (abstracted pest/disease early warning).
        + May require periodic calibration (a minigame or consumable CalibrationKitSO).
    - **Programmable Logic Controllers (PLCs) & Central Computer Systems:**
      * **PLC\_UnitSO (Equipment Item):** A placeable hardware unit. Players interact with its UI to define control logic.
      * **IF-THEN-ELSE Logic / Rule-Based Control:**
        + Players can create custom automation routines by linking sensor inputs to equipment outputs using conditional logic.
        + **UI:** A visual scripting interface or a structured rule-builder UI for the PLC.

*Example Rule:*

IF (RoomA\_TempSensor.Reading > 28°C) AND (RoomA\_FlowerCycle.IsActive == true)

THEN (RoomA\_AC\_Unit.SetTarget(25°C), RoomA\_AC\_Unit.TurnOn())

ELSE IF (RoomA\_TempSensor.Reading < 26°C)

THEN (RoomA\_AC\_Unit.TurnOff())

* + - * + PLCs can manage multiple inputs (sensors) and multiple outputs (equipment actuators).
      * **Environmental "Recipes" / Schedules:**
        + Players can define target environmental parameters (temp, humidity, CO2, light DLI) for each growth stage of a specific strain (learned through experimentation or from the AI Research Lab).
        + These recipes can be loaded into a PLC or a CentralFacilityComputerSO (a higher-tier controller).
        + The controller then dynamically adjusts connected HVAC, lighting, and CO2 systems to maintain the target recipe for plants in its assigned zone(s).
        + Schedules can include diurnal cycles (e.g., slightly lower night temperatures, gradual ramp-up/down of lights simulating dawn/dusk).
      * **PID Controllers (Advanced Concept):** For very precise control of certain parameters (e.g., nutrient solution temperature in RDWC), a Proportional-Integral-Derivative (PID) controller algorithm could be simulated within advanced PLCs. This offers smoother, more stable control than simple on/off logic, minimizing overshoot and oscillation.
    - **Data Logging & Graphing (Integrated with Controllers):**
      * Advanced controllers and the EnvironmentalMonitoringDashboardUI will provide detailed historical data logging and graphing for all connected sensors and equipment states, allowing players to analyze system performance and fine-tune their automation rules.
    - **Alerting & Notifications (Integrated):**
      * PLCs can be configured to send alerts to the player (via the main notification system) if parameters go out of a user-defined safe range, if equipment malfunctions, or if automation rules fail to achieve targets.
  + **Player Experience & Strategic Implications:**
    - Gradual transition from tedious manual control to designing and managing complex, self-regulating environmental systems.
    - Frees up player time from micromanagement to focus on higher-level strategy (breeding, market analysis, expansion).
    - Allows for much larger and more complex facilities to be managed effectively.
    - Designing efficient and robust automation logic becomes a new form of gameplay puzzle and optimization challenge.
    - Risk of poorly designed automation causing problems (e.g., conflicting rules, sensor misplacement leading to incorrect actions).
  + **C# Implementation (Full Vision):**
    - Sensor.cs (base class) with derived classes for each sensor type/tier, implementing ISensorDataProvider interface. NetworkedSensor.cs would additionally implement IDataNetworkedDevice.
    - Actuator.cs (base class for controllable equipment like lights, fans, pumps) implementing IActuator interface with methods like TurnOn(), TurnOff(), SetValue(float value).
    - PLC\_Controller.cs:
      * Manages a list of AutomationRule objects.
      * AutomationRule.cs: Contains conditions (references to sensors and comparison logic) and actions (references to actuators and commands).
      * Periodically evaluates rules and executes actions.
      * UI backend for the rule-builder interface.
    - EnvironmentalRecipeSO.cs: Stores target parameters per growth stage for a strain.
    - CentralFacilityComputer.cs: Higher-level controller that manages multiple PLCs or zones based on loaded recipes.
    - PIDController.cs (Utility Class): Implements PID algorithm if used.
  + **AI Tooling (Full Vision):**
    - Cursor AI: Assisting with the C# logic for the rule evaluation engine in PLC\_Controller.cs, the data structures for AutomationRule and EnvironmentalRecipeSO, and the implementation of various sensor/actuator classes.
    - AI for UI Design (Uizard, etc.): Potentially mocking up the visual scripting interface for the PLC rule builder.

Advanced automation is the capstone of the environmental control system, empowering players to achieve unprecedented levels of precision and scale in their cultivation endeavors, truly embodying the "high-tech" aspect of Project Chimera.

### 5.4. Nutrient Management: Feeding for Excellence

Proper nutrient management is a cornerstone of successful cultivation, directly impacting plant health, growth rates, yield, and the expression of valuable secondary metabolites like cannabinoids and terpenes. While the MVP introduces basic manual mixing and monitoring, the full vision encompasses a sophisticated simulation of plant nutrient requirements, diverse nutrient products, advanced delivery systems, and detailed analytical feedback.

* **MVP Recap (as per Part 4, Sec 4.3.4):**
  + **Manual Mixing:** Players use a UI to mix a few basic nutrient products (e.g., "Base Grow A/B," "Basic Bloom Booster," "CalMag").
  + **Manual Application:** Nutrient solutions applied with a basic watering can.
  + **Manual Monitoring:** Handheld EC/PPM and pH meters for checking mixed solutions and potentially runoff.
  + **Visual Feedback:** Generalized visual cues for severe nutrient imbalances (e.g., generic yellowing, tip burn).
  + **Exclusions:** No automated dosing, complex nutrient lines, detailed deficiency/toxicity modeling, advanced pH/EC management.
* **Full Vision: Precision Nutrition & Advanced Soil/Hydroponic Science:**

**5.4.1. Comprehensive Nutrient & Additive Library (NutrientProductSO):**  
The game will feature an extensive library of nutrient products, catering to different cultivation philosophies (synthetic, organic) and advanced feeding strategies.

* + **Base Nutrients (Synthetic):**
    - Multi-part liquid lines (e.g., Grow A/B/C, Bloom A/B, Micro). Each part (NutrientProductSO) will have a defined N-P-K ratio, micronutrient profile (Fe, Mn, Zn, Cu, B, Mo), and potentially secondary macronutrients (Ca, Mg, S).
    - Granular/Powdered synthetic nutrient salts (e.g., Calcium Nitrate, Potassium Phosphate, Magnesium Sulfate) for advanced players who want to mix custom base formulas from scratch. Requires more knowledge and careful calculation.
  + **Organic Nutrients & Amendments (for Living Soil - see 5.1.4):**
    - Bottled organic liquid nutrients (e.g., fish emulsion, liquid kelp, molasses).
    - Dry amendments for soil mixing or top dressing (KelpMealSO, AlfalfaMealSO, BoneMealSO, BloodMealSO, BatGuanoSO, RockPhosphateSO, LangbeiniteSO, AzomiteSO for trace minerals, WormCastingsSO). Each defines its NPK, micronutrient content, and slow-release characteristics.
  + **Supplements & Additives:**
    - **Root Enhancers/Stimulants:** Products designed to promote root growth and health (e.g., containing beneficial bacteria/fungi like mycorrhizae, vitamins, humic/fulvic acids).
    - **Bloom Boosters:** High P-K formulas for mid-to-late flower, often with specific micronutrients or biostimulants.
    - **Silica Supplements:** To strengthen cell walls, improve stress resistance, and increase stem rigidity.
    - **Enzyme Products:** To break down dead root matter and improve nutrient availability.
    - **Carbohydrate/Sugar Supplements:** Claimed to improve flavor/aroma or feed microbial life (more relevant for organic/living soil).
    - **Terpene Enhancers (Advanced/Speculative):** Products claiming to boost specific terpene production (effects would be subtle and GxE dependent).
    - **Flushing Agents (Late Flower):** Products designed to help "flush" excess salts from the growing medium before harvest.
  + **pH Adjusters:**
    - pH\_Up\_SolutionSO (e.g., Potassium Hydroxide based).
    - pH\_Down\_SolutionSO (e.g., Phosphoric Acid based for synthetics, Citric Acid based for organics).
  + **NutrientProductSO Properties:**
    - productID, productName, description, icon.
    - type (enum: BaseNutrient, OrganicAmendment, Supplement, pH\_Adjuster).
    - form (enum: Liquid, Powder, Granular).
    - NPK\_Ratio (Vector3 for N, P, K percentages).
    - microNutrientProfile (Dictionary<MicroNutrientType, float\_ppm\_contribution\_per\_unit>).
    - secondaryMacroProfile (Dictionary<SecondaryMacroType, float\_ppm\_contribution\_per\_unit>).
    - EC\_Contribution\_per\_unit (float).
    - pH\_Effect\_per\_unit (float - how much it tends to raise/lower pH).
    - cost\_per\_unit.
    - solubility (for powders/granulars).
    - releaseRate (enum: Fast, Medium, Slow - for organic amendments).
    - compatibleCultivationStyles (List<enum: Synthetic, Organic, Hydroponic, Soil>).
  + **C# Implementation:**
    - Extensive use of NutrientProductSO assets. Custom editor for easy data entry and validation.
    - NutrientDatabase.cs: A manager to hold references to all available nutrient products, allowing for easy lookup and filtering (e.g., in shop UI).

**5.4.2. Detailed Plant Nutrient Uptake & Deficiency/Toxicity Modeling:** The simulation will move beyond generic "nutrient level" to model the plant's need for, uptake of, and response to individual macro and micronutrients.

* + **Nutrient Requirements per Growth Stage & Strain:**
    - PlantStrainSO (or linked NutrientProfileSO) will define target optimal ranges or ratios for key nutrients (N, P, K, Ca, Mg) during different growth stages (seedling, veg, early/mid/late flower). These are genetic predispositions.
    - Actual uptake will be influenced by environmental factors (temperature, pH of medium, VPD affecting transpiration stream, root health) and plant size/vigor.
  + **Modeling Individual Nutrient Roles (Abstracted):**
    - **Nitrogen (N):** Crucial for vegetative growth, chlorophyll production, amino acids. Deficiency: yellowing of older leaves (chlorosis), stunted growth. Toxicity: very dark green leaves, weak stems, potential burning.
    - **Phosphorus (P):** Essential for root development, flowering, resin production, energy transfer. Deficiency: stunted growth, dark green/purplish leaves, poor flowering. Toxicity: can lock out other nutrients like zinc and iron.
    - **Potassium (K):** Important for enzyme activation, water regulation, flower/fruit development, overall plant vigor. Deficiency: yellowing/browning of leaf margins (older leaves first), weak stems, poor bud development. Toxicity: can lock out magnesium and calcium.
    - **Calcium (Ca):** Cell wall structure, enzyme function. Deficiency: new growth distortion (curling, spotting), blossom end rot in some plants (abstracted as bud issues). Immobile nutrient.
    - **Magnesium (Mg):** Central component of chlorophyll, enzyme activation. Deficiency: interveinal chlorosis (yellowing between veins) on older leaves. Mobile nutrient.
    - **Sulfur (S):** Amino acids, protein synthesis. Deficiency: general yellowing, often starting with younger leaves.
    - **Micronutrients (Fe, Mn, Zn, Cu, B, Mo):** Required in small amounts, but deficiencies cause specific symptoms. Modeled with less individual detail than macros, but overall "micronutrient balance" can be tracked. pH heavily affects their availability.
  + **Visual & Physiological Feedback:**
    - Specific, distinct visual cues (leaf discoloration patterns, growth distortions, burn marks) on the procedural plant models for moderate to severe deficiencies/toxicities of key nutrients (N, P, K, Ca, Mg). This requires advanced shader work and texture variations.
    - Impact on plant health stats, growth rate, photosynthetic efficiency, and ultimately yield and quality (cannabinoid/terpene expression via GxE).
  + **Nutrient Mobility:** The simulation will model nutrient mobility within the plant (e.g., N, P, K, Mg are mobile, so deficiency symptoms appear on older leaves first as the plant moves them to new growth. Ca, Fe, S are immobile, so symptoms appear on new growth). This aids player diagnosis.
  + **Nutrient Lockout:** Incorrect pH in the growing medium (too high or too low) will significantly reduce the plant's ability to absorb certain nutrients, even if they are present in the solution/soil. This "lockout" will be a key mechanic, forcing players to manage pH carefully.
  + **C# Implementation:**
    - PlantPhysiology.cs (on PlantInstance):
      * Tracks internal "pools" or current status for key nutrients (e.g., currentNitrogenLevel, targetNitrogenLevel).
      * Calculates nutrient demand based on growth stage, genetics, and current biomass.
      * Simulates uptake from the GrowingMedium.cs or HydroponicSystem.cs, influenced by medium pH, EC, root health.
      * If currentNutrientLevel deviates significantly from targetNutrientLevel for a sustained period, triggers deficiency/toxicity states.
    - NutrientImbalanceState.cs (part of PlantHealth FSM): Manages the application of visual effects and physiological penalties for specific imbalances.
    - GrowingMedium.cs / HydroponicSystem.cs: Tracks available nutrient concentrations and pH. The pH value will directly modify nutrient availability factors used by PlantPhysiology.cs.
    - NutrientDeficiencyVisualizer.cs: A script that applies specific shader effects or texture overlays to plant parts based on active nutrient imbalance states.

**5.4.3. Advanced Nutrient Solution Management & Analysis:** Players will have more sophisticated tools for mixing, analyzing, and managing their nutrient solutions, especially for hydroponics.

* + **Detailed Nutrient Mixing UI:**
    - Allows selection from the full library of NutrientProductSOs.
    - Calculates and displays not just total EC/PPM and pH, but also the estimated PPM concentrations of individual macronutrients (N, P, K, Ca, Mg, S) in the mixed solution. This requires NutrientProductSOs to have detailed compositional data.
    - "Recipe Book": Players can save their custom nutrient recipes (ingredient lists and amounts per gallon/liter) for different strains and growth stages.
  + **Water Quality Simulation:**
    - Starting water source (e.g., "Tap Water," "RO Water" from a ReverseOsmosisFilterSO equipment) will have a base EC and mineral profile. Tap water might contain chlorine/chloramine (negative effect if not filtered/aerated) or undesirable levels of certain minerals.
    - RO water is near 0 EC, providing a "clean slate" for nutrient mixing but requiring addition of CalMag.
  + **Nutrient Solution Stability & Temperature:**
    - In reservoirs, nutrient solutions can degrade over time or if temperature is too high (promoting algae/bacteria). Solution temperature (especially for DWC/RDWC) becomes a parameter to manage, potentially requiring WaterChillerSO equipment.
  + **Runoff & Leachate Analysis (Advanced):**
    - Players can collect runoff from pots or samples from hydroponic reservoirs.
    - An in-game "Lab Analysis Tool" (late-game equipment, or an NPC service) can provide a detailed report on the EC, pH, and PPMs of key nutrients in the submitted sample. This helps diagnose over/under feeding, nutrient imbalances, or salt buildup in the medium.
  + **Soil/Medium Testing (Advanced):**
    - Similar to runoff analysis, players might be able to take soil/coco samples and get a lab report on their nutrient content and pH. This is more relevant for living soil or re-amending coco.
  + **Foliar Feeding (Specialized Technique):**
    - Applying dilute nutrient solutions directly to plant leaves using a sprayer.
    - Used for quick correction of certain micronutrient deficiencies or applying specific biostimulants.
    - Risk of leaf burn if solution is too concentrated or applied in intense light.
    - Requires specific FoliarSprayRecipeSOs.
  + **C# Implementation:**
    - NutrientMixingUI.cs backend logic will need to perform complex calculations based on the detailed composition of all selected NutrientProductSOs.
    - WaterSourceSO.cs: Defines properties of tap vs. RO water.
    - Reservoir.cs: Tracks solution temperature, age, and potentially a "cleanliness" or "pathogen load" stat.
    - LabAnalysisToolController.cs: Simulates the analysis process (takes time, costs currency/resources) and generates a NutrientAnalysisReport data object.
    - FoliarFeedingController.cs: Handles the application and effects of foliar sprays.

**5.4.4. Automated Dosing & pH/EC Management (Integration with 5.3.4):** The full vision for nutrient management integrates seamlessly with advanced automation systems.

* + **Automated Dosing Pumps:** As described in 5.3.4, DosingPumpSO equipment items, controlled by PLCs or a CentralNutrientControllerSO, will automatically add precise amounts of concentrated nutrient stock solutions, pH adjusters, or supplements to reservoirs to maintain target EC, pH, and nutrient profiles.
  + **Inline Sensors:** InlineECSensorSO and InlinepHSensorSO provide real-time data from reservoirs or hydroponic systems to the automation controllers.
  + **Feedback Loops:** The PLC/Central Controller uses sensor data to make decisions:
    - If EC is low, add more base nutrients according to the current recipe's ratio.
    - If pH drifts, add pH Up/Down solution in small increments.
    - If a specific nutrient (if individual nutrient sensors become a very advanced feature) is low, dose that specific supplement.
  + **Player Configuration:** Player defines target EC/pH values and nutrient recipes in the controller's UI. They also need to "calibrate" the dosing pumps (how much a single dose affects EC/pH).
  + **Alarms & Safety:** Automation system alerts the player if it cannot maintain targets, if reservoirs of stock solutions run low, or if sensors malfunction.

The full nutrient management system aims to be a deep and engaging mini-game of chemistry, biology, and engineering. Players can choose simple manual methods or progress to highly sophisticated, automated precision nutrition, with each approach offering different levels of control, efficiency, and potential for optimizing plant performance.

### 5.5. Plant Health & Integrated Pest/Disease Management (IPM)

Maintaining optimal plant health and effectively managing pests and diseases are critical challenges in any cultivation endeavor. Project Chimera aims to simulate these aspects with increasing depth, from basic manual interventions in the MVP to sophisticated Integrated Pest Management (IPM) strategies in the full vision.

* **MVP Recap (as per Part 4, Sec 4.3.5):**
  + **Assessment:** Primarily visual plant health assessment.
  + **Threats:** 1-2 common, easily distinguishable pests (e.g., "Spider Mites," "Fungus Gnats") and 1-2 common diseases (e.g., "Powdery Mildew," "Root Rot" from overwatering).
  + **Scouting:** Basic manual tools: Magnifying Loupe, Sticky Traps. Simplified Microscope as an early Science skill tree unlock.
  + **Treatment:** 1-2 basic, manual organic treatments (e.g., "Neem Oil Spray," "Organic Fungicide Spray") using a hand sprayer.
  + **Guidance:** In-game "Plant Problems Guide" with simple descriptions, visual examples, and suggested MVP treatments.
  + **Impact:** Gradual health decline and negative visual changes if untreated; potential plant death.
  + **Exclusions:** No complex IPM, beneficials, systemic treatments, detailed pest/disease lifecycles, or strong environmental links to outbreaks (beyond very basic).
* **Full Vision: Sophisticated Plant Health Diagnostics & Proactive IPM Strategies:**

**5.5.1. Advanced Plant Health Modeling & Diagnostics:**  
The full vision expands plant health beyond a single stat into a multi-faceted system with detailed diagnostics.

* + **Comprehensive Health Metrics:**
    - OverallHealth (Percentage): A primary stat, but influenced by many sub-systems.
    - PhotosyntheticEfficiency (Percentage): Affected by leaf health, nutrient status, light, CO2. Directly impacts growth rate and biomass accumulation.
    - RootHealth (Percentage): Critical for nutrient/water uptake. Affected by medium aeration, moisture, temperature, pathogens (e.g., Pythium). Poor root health severely impacts the entire plant.
    - StressLevel (Percentage/Categorical): Accumulates from various environmental (heat, cold, drought, light burn), nutritional (deficiency/toxicity), and biological (pest/disease) stressors. High stress reduces growth, quality, and resistance.
    - DiseaseResistanceScore / PestResistanceScore: Base genetic values modified by current plant health and stress.
  + **Detailed Symptomology & Visual Feedback:**
    - Each specific nutrient deficiency/toxicity (as per 5.4.2), pest, and disease will have a unique and recognizable set of visual symptoms on the procedural plant models. This requires a rich library of shader effects, texture overlays, and potentially minor mesh deformations (e.g., leaf curling, gall formation).
    - Symptoms progress in severity if untreated (e.g., minor leaf spotting -> widespread lesions -> defoliation).
  + **Advanced Diagnostic Tools (Unlocked via Research/Skill Tree):**
    - **Digital Microscope (Upgraded):** Connects to an in-game computer interface. Provides higher magnification images of pests/pathogens, potentially with an AI-assisted (simulated) identification feature suggesting likely culprits from a broader database.
    - **Plant Sap Analysis Kit (Consumable):** Player takes a sap sample. The kit (used with a "Lab Bench" equipment item) gives a report on key nutrient levels *within the plant tissue*, helping to confirm suspected deficiencies/toxicities or identify "hidden hunger" before visual symptoms are severe.
    - **Leaf Tissue Analysis Service (NPC/Late-Game Lab):** Player sends leaf samples to an off-site (simulated) lab for a detailed report on nutrient content, pathogen presence (PCR test simulation for specific diseases), or even early stress markers. Takes time and costs significant currency.
    - **Environmental Data Correlation:** Advanced UI dashboards will allow players to overlay historical environmental data (temp, humidity, VPD) with plant health events or pest/disease outbreak timelines, helping them identify environmental triggers.

**5.5.2. Expanded Roster of Pests & Diseases (PestSO, DiseaseSO):** The game will feature a wider variety of common cannabis pests and diseases, each with unique characteristics.

* + **Pests:**
    - **Spider Mites (Tetranychidae):** Tiny, cause stippling on leaves, fine webbing in severe infestations. Thrive in hot, dry conditions.
    - **Fungus Gnats (Sciaridae):** Small, dark flies. Adults are annoying; larvae in soil can damage roots, especially in seedlings. Thrive in overly moist media.
    - **Aphids (Aphidoidea):** Small, soft-bodied insects, often on new growth, suck sap, excrete honeydew (leading to sooty mold).
    - **Thrips (Thysanoptera):** Tiny, slender insects, cause silvery/rasped patches on leaves. Can transmit viruses.
    - **Whiteflies (Aleyrodidae):** Small, white, winged insects, underside of leaves, suck sap.
    - **Broad Mites / Russet Mites (Tarsonemidae):** Microscopic, cause distorted, twisted, shiny new growth. Very damaging.
    - **Caterpillars / Inchworms (Lepidoptera larvae):** Chew holes in leaves and buds. More common in outdoor/greenhouse settings.
    - **Root Aphids:** Attack roots, difficult to detect, cause wilting/stunting.
  + **Diseases (Fungal, Bacterial, Viral - Abstracted):**
    - **Powdery Mildew (Erysiphales):** White, powdery spots on leaves/stems. Thrives in high humidity, moderate temps, poor airflow.
    - **Botrytis / Bud Rot / Grey Mold (Botrytis cinerea):** Affects dense buds, especially in cool, damp, stagnant conditions. Devastating late in flower.
    - **Root Rot (Pythium, Fusarium, etc.):** Caused by various soil/water-borne pathogens. Roots become brown, slimy. Plant wilts, stunts. Common in overwatered media or poorly oxygenated hydroponics.
    - **Leaf Septoria / Yellow Leaf Spot:** Fungal, causes yellow/brown spots with dark borders on leaves.
    - **Downy Mildew (Peronosporaceae):** Similar to powdery mildew but often on underside of leaves, can cause yellow lesions on top. Needs high humidity.
    - **Verticillium Wilt:** Soil-borne fungus, causes wilting on one side of plant/branches, vascular discoloration.
    - **Tobacco Mosaic Virus (TMV) / Hop Latent Viroid (HpLVd) (Abstracted "Viral/Viroid Infection"):** Can cause mottling, distortion, stunting, reduced vigor/potency. Spread via contact, tools, pests. No direct "cure" in-game, focus on prevention and culling infected plants.
  + **PestSO / DiseaseSO Properties:**
    - ID, Name, Description, SymptomsList (text and links to visual cues).
    - OptimalConditionsForOutbreak (e.g., temp range, humidity range, links to plant stress factors).
    - SpreadMechanism (enum: Airborne, Waterborne, Contact, VectorPest).
    - DamageRate (how quickly it affects plant health/quality).
    - TreatmentEffectiveness (Dictionary<TreatmentItemSO, float\_effectiveness\_modifier>).
    - GeneticResistanceFactor (base susceptibility, modified by plant genetics).
    - VisualCuePrefabs/ShaderEffects (links to assets for showing the pest/disease on plants).

**5.5.3. Integrated Pest Management (IPM) Strategies:** IPM is a holistic approach focusing on prevention, monitoring, and using the least harmful control methods first. This becomes a core strategic layer.

* + **Cultural Controls (Prevention):**
    - **Sanitation:** Regularly "cleaning" the grow room (an abstracted player action or a stat for the room) reduces pest/pathogen reservoirs. Sterilizing pots and tools between grows.
    - **Environmental Management:** Maintaining optimal temperature, humidity, and airflow to create conditions less favorable for specific pests/diseases (e.g., good airflow to prevent powdery mildew and botrytis). This links directly to the microclimate system (5.3.3).
    - **Resistant Cultivars:** Players breeding or selecting strains with higher genetic resistance (PlantStrainSO.pestDiseaseResistanceScores) to prevalent threats in their facility.
    - **Quarantine Procedures:** A dedicated "Quarantine Room/Zone" where new plants (from outside sources or unverified clones) are held and monitored for a period before being introduced to main grow areas. This is unlocked via research.
    - **Air Filtration (HVAC):** High-quality intake filters on the HVAC system (see 5.3.1) can reduce airborne spore/pest entry.
  + **Biological Controls (Beneficial Organisms - Post-MVP):**
    - **Predatory Mites (e.g., *Phytoseiulus persimilis* for spider mites, *Amblyseius californicus*):** Consumable items (BeneficialOrganismSO) that player purchases and releases into grow rooms. They actively hunt and consume specific pest mites. Their population dynamics (establishment, reproduction, decline if prey is scarce) are simulated simply.
    - **Predatory Insects (e.g., Ladybugs for aphids, Green Lacewing larvae):** Similar release mechanic.
    - **Beneficial Nematodes (e.g., *Steinernema feltiae* for fungus gnat larvae):** Applied to soil/medium.
    - **Microbial Inoculants (e.g., *Bacillus thuringiensis* - Bt for caterpillars, *Trichoderma* species for root health/disease suppression):** Sprayed or added to medium.
    - **Considerations:** Beneficials may have specific environmental requirements themselves. Using broad-spectrum chemical pesticides will kill them.
  + **Physical/Mechanical Controls:**
    - **Sticky Traps (MVP carry-over):** Continued use for monitoring and mass trapping of flying insects.
    - **Hand Removal (for larger pests like caterpillars):** A player interaction.
    - **Pruning Infected Parts:** Carefully removing and "disposing" of diseased leaves or buds to prevent spread (with a small risk of spreading spores if not done carefully - skill check or tool dependent).
    - **Diatomaceous Earth (Consumable):** A desiccant powder applied to soil surface or dusted on plants to control soft-bodied insects.
  + **Chemical Controls (Last Resort, with consequences):**
    - **Organic Pesticides/Fungicides (MVP carry-over):** Neem oil, insecticidal soaps, potassium bicarbonate, sulfur sprays. Generally safer but may require more frequent application.
    - **Synthetic Pesticides/Fungicides (Advanced Unlock, High Risk/Reward):** More potent and potentially systemic options.
      * **Consequences:**
        + May harm or kill beneficial organisms.
        + Potential for "Pesticide Residue" on final product if applied too close to harvest, leading to quality penalties, market rejection, or reputation loss.
        + Risk of pests/diseases developing resistance if the same chemicals are overused (a simulated "resistance buildup" mechanic for specific pest/pathogen populations in a facility).
      * Requires careful adherence to (simulated) pre-harvest intervals (PHI).
  + **IPM Strategy UI:** A dedicated UI panel where players can:
    - View current pest/disease threats across their facility.
    - See logs of scouting activities and treatment applications.
    - Plan preventative schedules (e.g., "Release predatory mites in Flower Room 1 every 2 weeks").
    - Access information on different IPM methods and their pros/cons.

**5.5.4. Environmental Influence on Outbreaks & Spread:** The microclimate system (5.3.3) will directly influence pest/disease risk.

* + **Favorable Conditions:** Each PestSO / DiseaseSO will have defined optimal environmental conditions (temperature, humidity ranges) for its reproduction and spread. If a grow room's microclimate falls into these ranges for extended periods, the probability of an outbreak or rapid spread increases significantly.
  + **Airflow & Spore Dispersal:** Poor airflow (stagnant air pockets in the microclimate simulation) increases risk of fungal spore settlement and germination. Fans and HVAC systems that create good air exchange reduce this risk.
  + **Watering Practices & Root Zone Health:** Overwatering or poorly drained media create anaerobic conditions favorable for root rot pathogens. The hydroponic system's water temperature and oxygenation directly impact root health and Pythium risk.
  + **Contamination Vectors:**
    - **Tools:** Using non-sterilized pruning tools on an infected plant and then a healthy one can spread diseases (a small probability). A "Tool Sterilization" action or consumable (IsopropylAlcoholWipesSO) can mitigate this.
    - **Player Movement (Abstracted):** Moving directly from an infested room to a clean room without a "decontamination" step (e.g., changing conceptual coveralls, an action at the room entrance) might carry a small risk of transferring pests/spores.
    - **New Plants:** Clones or seeds from external sources (NPCs) can introduce new pests/diseases if not quarantined.

**5.5.5. C# Implementation (Full Vision for Plant Health & IPM):**

* + PlantHealthManager.cs: A central system that might oversee outbreak probabilities, track overall facility pest/disease pressure, and manage the progression of afflictions across multiple plants.
  + AfflictionInstance.cs (on PlantInstance): Represents an active pest infestation or disease infection on a specific plant. Manages its severity, visual state, and response to treatments. Interacts with PlantPhysiology.cs to apply damage.
  + IPMController.cs: Handles player IPM actions, application of treatments (chemical, biological), and manages the state/population of beneficial organisms in a room.
  + EnvironmentalFactorEvaluator.cs: A module queried by PlantHealthManager that assesses current microclimate conditions in each room (from RoomEnvironmentController.cs) and determines risk modifiers for specific pests/diseases based on their PestSO/DiseaseSO optimal conditions.
  + GeneticResistanceModule.cs: Calculates a plant's current resistance to a specific pest/disease based on its PlantStrainSO.pestDiseaseResistanceScores and current health/stress level.
  + Extensive UI for diagnostic tools, IPM strategy planning, and detailed plant health readouts.
  + Shaders and visual effect systems to render a wide variety of specific pest/disease symptoms on plants.

**5.AI Tooling (Full Vision for Plant Health & IPM):**

* + Cursor AI: Logic for AfflictionInstance state progression, IPMController action handling, risk calculation in EnvironmentalFactorEvaluator. Boilerplate for the many PestSO/DiseaseSO/TreatmentItemSO/BeneficialOrganismSO assets.
  + AI for Text Generation: Assisting with detailed descriptions of symptoms, pest/disease lifecycles, and IPM strategy tips for the in-game guides and diagnostic tools.
  + AI for Image Generation: Creating diverse visual examples of pest/disease symptoms for the guides, or icons for various treatments and beneficials.

The full vision for plant health and IPM transforms it from a simple reactive system into a deep, strategic layer of gameplay. Players must become knowledgeable diagnosticians and proactive managers, using a combination of environmental control, genetic selection, biological agents, and targeted treatments to keep their high-value crops healthy and productive.

### 5.6. Time Mechanics: Controlling the Flow of Growth

The time mechanic is a fundamental system in any simulation game dealing with long biological processes. For Project Chimera, it's crucial for balancing realism with engaging gameplay, allowing players to observe rapid growth when desired and manage their game around real-life schedules. The MVP establishes a robust foundation for this, which the full vision refines and potentially expands. (Doc1, Sec II.F).

* **MVP Recap (as per Part 4, Sec 4.3.6):**
  + **Active Time Scales:** 0.5x, 1x (baseline: 1 in-game week = 1 real-world hour), 2x, 4x, 8x, and a "Real-time" 1:1 option.
  + **"Transition Inertia":** Mandatory lock-in and transition delay when changing speeds, preventing trivial switching.
  + **Offline Progression:** Full player agency to select any active time scale or PAUSE the game entirely while offline.
  + **Catch-Up:** "Catch-Up Visualization" and "Facility Status Report" upon login after offline progression.
  + **Subtle Nuances:** Slower speeds *may* slightly increase max quality potential; faster speeds *might* slightly increase minor stressor probability if not well-managed.
  + **Exclusions:** Complex event scheduling tied to time scales.
* **Full Vision: Strategic Time Management & Deeper Integration:**

**5.6.1. Refined Active Time Acceleration & Control:**  
The core MVP time scales provide a good range. The full vision focuses on refining their impact and the player's interaction with them.

* + **UI/UX for Time Control:**
    - **Clear Visual Feedback:** The UI must always clearly display the current time scale, the progress of any "Transition Inertia" ramp-up/down, and the duration of any lockout from changing speeds.
    - **"Commitment Warning" Pop-up:** Before a speed change is finalized (triggering Transition Inertia), a more detailed pop-up (as per Doc1, Sec II.F) will summarize:
      * Estimated real-world time for 1 in-game day/week at the new speed.
      * Potential increased frequency of manual tasks (if automation is low).
      * Reminder of the lock-in period.
      * Brief note on the subtle risk/reward nuances associated with the new speed.
    - **Customizable Hotkeys:** Allow players to assign hotkeys for frequently used time speeds and pause.
  + **"Smart Pause" Functionality (Potential Enhancement):**
    - **Concept:** An optional setting where the game automatically pauses if certain critical events occur while time is highly accelerated (e.g., major equipment failure, critical plant health alert, important NPC communication).
    - **Player Configuration:** Players can toggle this on/off and potentially select which event severities trigger the auto-pause.
    - **Benefit:** Prevents players from missing catastrophic events if they step away briefly while the game is running fast.

**5.6.2. Enhanced "Transition Inertia" System:** The core concept from MVP is sound. The full vision ensures it feels fair and strategically impactful.

* + **Dynamic Transition Duration Calculation:**
    - The duration of the speed ramp-up/down and the associated lockout period could be more dynamically calculated, as suggested in Doc2 (Sec VIII.A), e.g., "a percentage of the real-world time for one in-game day at the slower of the two speeds involved in the transition."
    - This makes larger jumps in speed require a more significant time commitment for the transition itself.
  + **Visual & Auditory Cues for Transition:**
    - Subtle visual effects (e.g., a slight motion blur that intensifies/fades during ramp, screen edge vignette) or auditory cues (e.g., a gentle winding up/down sound effect) could accompany the speed transition to make it feel more immersive and less like an abrupt digital switch.
  + **No Exploits:** Rigorously test to ensure players cannot find ways to bypass or exploit the Transition Inertia system (e.g., by quick saving/loading, or through specific UI interactions).

**5.6.3. Deeper Integration of Time Scale with Game Systems (Subtle Risk/Reward Variables - Full Detail):** The MVP's subtle risk/reward nuances (Doc1, Sec II.F) are expanded and made more integral, though still subtle enough not to force a single playstyle.

* + **Impact on Genetic Expression Quality (Refined):**
    - **Mechanism:** Slower time scales (e.g., 0.5x, 1x) might allow for a slightly more "meticulous" GxE interaction simulation, providing a small bonus (e.g., +0.5% to +2% to the *maximum achievable potential* for cannabinoid/terpene expression if all other conditions are perfect). This represents the idea that slower, more stable processes can sometimes yield higher quality.
    - **Balancing:** This bonus must be small enough that it doesn't make faster speeds feel strictly inferior, especially once automation handles consistency. It's a slight edge for patient players, not a mandatory path.
  + **Impact on Stressor Probability & Severity (Refined):**
    - **Mechanism:** Faster time scales (e.g., 4x, 8x) could introduce a *marginal* increase (e.g., +1% to +5% base probability, or slightly faster progression) for minor stressors like:
      * Small pest outbreaks (if preventative IPM isn't perfect).
      * Minor equipment glitches (e.g., a light flickers, a pump temporarily loses pressure – requiring a quick manual reset).
      * Slightly faster depletion of very short-lived consumables (e.g., CO2 in a poorly sealed room).
    - **Rationale:** Represents the idea that managing complex systems at high speed inherently carries more risk if not perfectly automated and monitored. It's easier for small issues to escalate if the player isn't reacting as quickly in real-time relative to game-time.
    - **Mitigation:** Excellent automation (unlocked post-MVP) should largely negate these increased risks, making faster speeds viable and desirable for well-managed late-game facilities.
  + **Impact on Research & Construction (Optional):**
    - **Research Speed:** Could be a direct multiplier (e.g., at 2x speed, research progresses twice as fast in game-time, meaning it takes half the real-world time). This is the most straightforward approach.
    - **Construction Speed:** Similar to research, build times for facility components or equipment installation could scale directly with game time.
  + **Impact on NPC Behavior & Market Dynamics (Post-MVP):**
    - If NPCs have schedules or the market has timed events/fluctuations, these will naturally occur more frequently in real-time at higher game speeds.
    - The AI for NPC competitors (if implemented) might make decisions or take actions based on game-time progression, meaning their "turns" come faster at accelerated speeds.
  + **Communicating these Effects:**
    - Subtle hints from ADA or tooltips on the time control UI.
    - Potentially, advanced data logs could show correlations if players are analytical (e.g., "Analysis of last 5 grow cycles indicates a 1.2% average higher terpene retention when operating at 1x speed vs 4x speed for Strain Y under similar conditions"). This is very late-game.

**5.6.4. Robust Offline Progression & Catch-Up (Full Detail):** The MVP system is good. The full vision ensures it's seamless and informative.

* + **Saving Offline Progression Choice:** The player's selected offline time scale (or "Paused") is saved reliably with their game session.
  + **"Fast-Forward" Simulation Loop (Upon Login):**
    - **Accuracy vs. Performance:** This loop needs to be carefully designed. It cannot re-simulate every single frame or every minor event with full fidelity, as that could take too long for players who were offline for many real-world days at a high game speed.
    - **Key System Updates:** The loop will focus on updating the most critical systems in discrete, larger time steps:
      * Plant Growth: Apply accumulated growth based on average environmental conditions during offline period and GxE.
      * Resource Consumption: Deduct total water, nutrients, power based on active equipment and duration.
      * Consumable Depletion: Update inventories.
      * Research/Construction Timers: Advance progress.
      * Contract Deadlines: Check for expirations.
      * **Probabilistic Event Checks:** For events like equipment malfunctions or pest outbreaks, instead of simulating every chance, the system might roll a cumulative probability based on the offline duration and risk factors (e.g., "Over 7 offline days, with moderate pest pressure, there was a 30% chance of a minor spider mite outbreak in Room B. Roll: Success/Failure."). If an event "occurred," its consequences are applied.
    - **State Prioritization:** The simulation must ensure that critical states (e.g., a plant dying due to resource depletion) are accurately captured.
  + **"Catch-Up Visualization" (Enhanced):**
    - **More Detail (Optional):** Beyond a simple time-lapse of UI bars, it could show highly accelerated, iconic visual changes in the main facility view (e.g., plants visibly growing in fast-motion, resource icons depleting, day/night cycling rapidly). This requires a performant way to represent these changes without full rendering.
    - **Skippable but Informative:** Player can skip the visualization if they wish, but it should offer a good summary if watched.
  + **"Facility Status Report" (Comprehensive):**
    - **Categorized Information:** Clearly structured with sections for:
      * **Overall Summary:** Time elapsed, chosen offline speed.
      * **Facility Resources:** Starting/Ending levels for Currency, Water, Power (if tracked as a resource like generator fuel), key Nutrients.
      * **Crop Progress:** For each active grow room/batch: Stage advancement, number of plants harvested (if any), yield obtained.
      * **Genetic Developments:** Seeds produced (if automated breeding was somehow active – unlikely for MVP, but placeholder), clones rooted.
      * **Research & Construction:** Projects completed, progress on ongoing ones.
      * **Economic Activity:** Contracts fulfilled/failed, income generated, major expenses.
      * **Significant Events:** List of key occurrences:
        + Critical Alerts (e.g., "Power outage in Sector Alpha for 3 in-game hours").
        + Equipment Malfunctions.
        + Pest/Disease Outbreaks (if occurred probabilistically).
        + Plants reaching harvestable state or dying.
      * **Actionable Items:** Highlight any immediate issues requiring player attention (e.g., "Reservoir Omega is empty," "Contract X is due in 2 in-game hours").
    - **Clickable Links:** Where possible, items in the report could be clickable, taking the player directly to the relevant UI panel or facility area.
  + **C# Implementation (Full Vision for Offline):**
    - OfflineProgressionManager.cs:
      * Calculates total game-time to simulate based on real\_time\_elapsed \* chosen\_offline\_timescale.
      * Divides this into manageable simulation "ticks" (e.g., 1 in-game hour or 6 in-game hours per tick).
      * In each tick, calls specialized SimulateOfflineTick(float offlineDeltaTime) methods on key managers (CultivationManager, EconomyManager, EnvironmentManager, ProgressionManager). These methods would contain simplified, aggregated logic suitable for fast-forward simulation.
      * Logs all significant events to populate the FacilityStatusReportData object.
    - FacilityStatusReportGenerator.cs: Takes the FacilityStatusReportData and formats it for UI display.
    - The "Catch-Up Visualization" would likely be driven by a separate script that rapidly tweens UI elements or plays pre-defined iconic animations based on the summary data from the offline simulation.

**5.6.5. Time-Based Events & Scheduling (Post-MVP Advanced):** Beyond player-controlled acceleration, the full game world could feature time-based events and allow players to schedule future actions.

* + **Scheduled Game World Events:**
    - **Market Fluctuations:** Certain commodity prices (e.g., specific nutrients, base building materials) might have weekly or seasonal trends. Special NPC buyers or contract opportunities might only be available on certain in-game days or during specific "market seasons."
    - **NPC Schedules (If Complex NPCs are Added):** Key NPCs might have availability schedules, affecting when players can interact with them for high-value trades or unique quests.
    - **Weather Patterns (If Outdoor Growing is Added):** Predictable (to some extent) seasonal weather patterns or occasional random weather events (heatwave, cold snap) that affect outdoor or greenhouse cultivation.
  + **Player-Schedulable Actions (Advanced Automation):**
    - **Concept:** A late-game upgrade to PLCs or the Central Facility Computer allowing players to schedule specific actions or changes to automation recipes at future in-game dates/times.
    - *Example:* "On 7/15 at 08:00, switch Flower Room 3 from Veg Recipe to Early Flower Recipe." "Automatically start flushing nutrients for Batch B three days before their scheduled harvest date."
    - **UI:** A calendar-based scheduling interface.
    - **Benefit:** Allows for very long-term planning and automation of entire grow cycles.
  + **C# Implementation (Full Vision for Scheduling):**
    - WorldEventManager.cs: Manages a timeline of scheduled game world events, triggering them when the in-game time matches. Events could be defined in WorldEventSOs.
    - PlayerScheduler.cs: Allows players to create ScheduledPlayerActionSOs. The TimeManager or a dedicated scheduler checks for due actions each game tick.

The full vision for Time Mechanics aims to provide a system that is not only a convenience for managing long processes but also a strategic layer with subtle depths and, eventually, a framework for interacting with a more dynamic and time-aware game world. The MVP's robust foundation for speed control and offline progression is critical for this evolution.

### 5.7. Facility Construction & Management: Building the Ideal Grow Op

This system empowers players to design, build, expand, and optimize their cultivation facilities, progressing from a humble residential setup to sprawling, high-tech warehouses. It's a core pillar that blends creative design with engineering challenges.

* **MVP Recap (as per Part 4, Sec 4.3.7 & relevant parts of Doc1, Sec VI):**
  + **Sandbox:** Residential House with predefined layout and unlockable rooms. Warehouse conceptually unlocked as a future goal, perhaps with its basic empty shell accessible but not for detailed build-out.
  + **Construction:** Grid-based placement of basic equipment within existing rooms of the House. No custom room construction by player in MVP (they unlock pre-existing rooms).
  + **Utilities:** Abstracted power/water in the House (e.g., wall outlets, sink taps). No manual 3D utility routing or X-Ray view in MVP.
  + **Zoning:** Rudimentary designation of entire rooms for specific functions (e.g., "Veg Room," "Flower Room") via UI.
* **Full Vision: Custom Free-Form Construction, Intricate Utility Engineering, & Strategic Layout Optimization:**

**5.7.1. Advanced Sandbox Environments & Expansion:**Beyond the Residential House, the Warehouse and potentially other future facility types become fully realized sandboxes.

* + **The Warehouse (Full Functionality - Post-MVP):**
    - **Vast Empty Shell:** Players start with a large, open concrete floor, supporting pillars, and a basic roof structure (as per Doc1, Sec VI.A). The scale is significantly larger than the House.
    - **Complete Construction Freedom:** Players can design and build custom interior rooms from scratch using a wide array of structural components (walls, floors, ceilings, doors, windows).
    - **Multi-Story Construction:** Ability to build vertically within the Warehouse's height limits, adding new floors, stairs, and elevators/lifts.
    - **Specialized Room Types:** Players will design and dedicate rooms for: Seedling/Cloning, Vegetative Growth (multiple stages), Flowering (multiple rooms for perpetual harvest), Drying, Curing, Trimming/Processing, Genetics Lab, AI Research Lab, Tissue Culture Lab, Nutrient Mixing/Storage, Utility/Maintenance, Office/Breakroom.
  + **Future Facility Types (Long-Term Expansions):**
    - **Automated Greenhouses:** Large-scale structures with transparent roofs (simulating sunlight as a light source, modified by glazing material and external weather if implemented). Different glazing materials (GlazingSO) could offer varying light transmission, insulation, and UV filtering. Requires managing ventilation for heat buildup.
    - **Outdoor Grows (Very Advanced):** Requires simulation of external weather (seasons, rain, temperature, wind, sunlight hours/intensity), soil plots, pest/disease pressures from the natural environment. A completely different set of challenges.
    - **Underground Bunkers / Specialized Research Facilities:** Thematic variations offering unique constraints or benefits.
  + **C# Implementation:**
    - FacilityManager.cs: Manages all player-owned facilities.
    - FacilityData.cs: Stores the state of each facility (layout, rooms, equipment).
    - Scene management for loading/unloading different facility types or large sections of a massive warehouse.

**5.7.2. Sophisticated Grid-Based & Free-Form Construction System:** Players gain much greater control over how they build and shape their spaces.

* + **Enhanced Grid System (Doc1, Sec VI.B):**
    - **Fine Granularity:** Retains the 1-foot (or metric equivalent like 0.3m) fundamental grid unit for precision.
    - **Advanced Snapping:** Snap to grid lines, intersections, mid-points, vertices of existing objects, and angles (e.g., 45°, 90°). Toggleable for freeform adjustments.
    - **Vertical Grid:** For multi-story construction.
  + **Structural Components Library (StructuralElementSO):**
    - **Walls:** Various types: Drywall, Insulated Panels (different R-values affecting heat transfer - Doc1, Table 2), Concrete, Reinforced Walls (security/structural integrity for very large builds). Different visual styles.
    - **Floors/Ceilings:** Concrete Slabs, Raised Access Flooring (for under-floor utilities), Steel Grating, Insulated Ceiling Panels.
    - **Doors:** Standard Interior, Airtight/Sealed Doors (for better zone isolation), Fire-Rated Doors, Large Industrial Roller Doors. Different access control levels (manual, keycard - if security/staff mechanics are added).
    - **Windows:** Standard, Double/Triple Glazed (insulation), UV-filtering options (for specific research or to reduce degradation of stored materials).
    - **Support Structures:** Pillars, I-beams for supporting large rooms or upper floors.
  + **Placement & Building Mechanics:**
    - **Blueprint Mode:** A top-down or isometric view for laying out walls, rooms, and large equipment. Similar to architecture software.
    - **First-Person/Third-Person Detail Mode:** For placing smaller equipment, routing utilities precisely, and inspecting the facility up close.
    - **"Drag to Build" Walls/Floors:** Click and drag to quickly draw out rectangular rooms or floor sections.
    - **Modular & Prefab-Based:** Most structural elements are modular prefabs that snap together.
    - **Validation Logic:** Prevents impossible placements (e.g., walls floating in mid-air without support, overlapping structures). Checks for structural integrity for large spans or multi-story builds (abstracted).
  + **Demolition & Modification:** Easy tools for demolishing existing structures and reconfiguring layouts (with resource recovery, potentially partial).
  + **C# Implementation:**
    - ConstructionController.cs: Manages player input for building, placement validation, and interaction with the grid system.
    - GridSystem.cs: Handles the underlying 3D grid logic, snapping calculations.
    - StructuralElement.cs (MonoBehaviour on placed objects): Stores its type (StructuralElementSO), health/integrity, and connections to other elements.
    - FacilityLayoutData.cs: Stores the 3D array or list of all placed structural elements for a facility.

**5.7.3. Full 3D Manual Utility Routing (Reiteration & Expansion - see 5.3.1):** This is a core engineering challenge, moving from MVP's abstracted utilities to detailed, player-designed networks.

* + **Visual & Functional Depth:** Players will see and interact with every pipe, wire, and duct they place. The functionality of equipment depends entirely on correct and adequate utility connections.
  + **Capacity & Bottlenecks:**
    - Pipe diameters limit water/nutrient flow. Too many devices on a small pipe reduce pressure/flow at the end of the line.
    - Wire gauges limit amperage. Overloading a wire can cause it to (figuratively) "burn out" or trip a breaker.
    - Duct sizes limit airflow. Undersized ducts for a large room or powerful fan will be inefficient.
  + **Maintenance & Repair (Post-MVP):**
    - Utility components (especially pumps, generators, complex controllers) can degrade over time and require maintenance (a consumable MaintenanceKitSO or a timed player interaction).
    - Chance of random failures (leaks in pipes, shorts in wires, fan motor burnout) increasing with age or if components are stressed (e.g., overloaded circuits, pumps running dry). Failures require diagnosis (using X-Ray view) and repair.
  + **C# Implementation:** As detailed in 5.3.1 (Graph-based networks, UtilityNetworkManagers). The maintenance/failure logic would add states to PlaceableUtilityComponent.cs and a MaintenanceManager.cs to track component health and trigger repair tasks/events.

**5.7.4. Advanced Zoning & Layout Optimization (Full Detail):** Strategic facility layout and zoning become crucial for efficiency, environmental control, and risk management. (Doc1, Sec VI.C).

* + **Granular Zoning UI:**
    - Players use a "Zoning Tool" in blueprint mode to draw and assign zones within their facility.
    - Zones can be designated for:
      * Seedling & Cloning (high humidity, specific light spectrum).
      * Vegetative Growth (specific photoperiod, nutrient recipes, environmental targets).
      * Flowering (different photoperiod, recipes, environmental targets, potentially higher CO2).
      * Mother Plant Room (stable environment, specific lighting to prevent flowering).
      * Drying Room (low light, specific temp/humidity/airflow targets).
      * Curing Area (stable temp, space for many jars).
      * Trimming & Processing Room.
      * Genetics Lab / AI Research Lab / Tissue Culture Lab (cleanliness stat, specialized equipment).
      * Nutrient Mixing & Storage.
      * Utility & Maintenance Access.
      * Quarantine Zone (for new plants).
      * Office / Staff Breakroom (if staff are added).
  + **Gameplay Effects of Zoning:**
    - **Environmental Isolation:** Well-sealed zones (using airtight doors, good wall construction) are easier to maintain distinct microclimates in. Poor sealing leads to environmental bleed between zones.
    - **Equipment Placement Rules:** Some advanced or specialized equipment might only be placeable or function optimally within appropriately designated zones (e.g., a "Tissue Culture Flow Hood" only in a "Tissue Culture Lab" zone).
    - **Workflow Optimization:** Players need to plan layouts for efficient movement of plants (seedlings -> veg -> flower -> dry -> cure -> trim), materials (nutrients to mixing stations, then to reservoirs), and potentially staff. Inefficient layouts increase travel time (abstracted as reduced efficiency or higher labor costs if staff are simulated).
    - **Contamination Control:**
      * Diseases or pests originating in one zone can spread to adjacent zones.
      * "Clean" zones like Tissue Culture Labs or Cloning Rooms might require stricter protocols (e.g., airlocks, foot baths - abstracted as higher construction cost or special door types) to prevent contamination. Contamination events reduce success rates or cause losses.
      * Quarantine zones help prevent introducing pests/diseases from new genetic stock into the main facility.
    - **Targeted Automation:** PLCs and Central Facility Computers (see 5.3.4) can be assigned to manage specific zones, applying tailored environmental recipes and automation rules.
  + **Visual Analysis Tools for Layout Optimization (Post-MVP):**
    - **Environmental Heatmaps (Overlay):** In blueprint or X-Ray view, toggle overlays showing heatmaps for temperature, humidity, light intensity, or airflow across the facility, helping to identify problem areas or inconsistencies.
    - **Workflow Pathing Visualization (Abstracted):** A tool that shows common paths for plant/material movement, highlighting potential bottlenecks or overly long travel distances in the layout.
    - **Contamination Spread Risk Overlay:** Visualizes potential pathways for pest/disease spread based on zone adjacency, airflow, and traffic patterns.
  + **C# Implementation:**
    - Zone.cs: A data object representing a defined area, storing its type, assigned environmental recipe (if any), and links to the RoomEnvironmentControllers or cells it encompasses.
    - ZoningManager.cs: Manages all defined zones, provides data to other systems (e.g., automation controllers, contamination simulator).
    - UI scripts for the zoning tool and the visual analysis overlays.
    - ContaminationSimulator.cs: A module that models the probability of pest/disease spread between zones based on proximity, airflow links, sanitation levels, and quarantine effectiveness.

Strategic zoning and layout directly impact operational efficiency, resource consumption, risk management, and the ability to scale effectively. It turns facility design into a deep strategic puzzle.

**5.7.5. C# Implementation (Overall for Facility Construction & Management):**

* + **Managers:** FacilityManager.cs, ConstructionController.cs, GridSystem.cs, ZoningManager.cs, UtilityNetworkManager.cs (per utility type).
  + **Data Objects:** StructuralElementSO.cs, PlaceableEquipmentSO.cs, ZoneData.cs, FacilityLayoutSaveData.cs.
  + **MonoBehaviours:** StructuralElementInstance.cs, PlaceableEquipmentInstance.cs, RoomInstance.cs.
  + Heavy reliance on Unity's physics for raycasting (placement, selection) and potentially basic collision for structural integrity checks (abstracted).
  + Extensive UI for blueprint mode, construction tools, zoning tools, X-Ray view, and visual analysis overlays.

**5.AI Tooling (Full Vision for Facility Construction & Management):**

* + Cursor AI: Assisting with the complex C# logic for the grid system, placement validation, utility network graph algorithms, zoning management, and the backend for the visual analysis tools.
  + AI for 3D Assets (Rodin, Sloyd, etc.): Generating base meshes for the wide variety of structural components, utility parts, and specialized room equipment, all requiring human optimization.
  + AI for Concept Art: Visualizing different facility layouts, room designs, and the aesthetic of advanced equipment.
  + Unity AI (Behavior Trees - Experimental): If staff management is ever added, AI could control NPC staff performing tasks based on facility layout and assigned zones.

The full vision for Facility Construction & Management provides players with immense creative freedom and deep strategic challenges, making the facility itself a core "character" in the game that the player designs, builds, and evolves.

### 5.8. Post-Harvest Processing: From Flower to Finished Product

The journey of a cannabis plant doesn't end at harvest. The post-harvest phase, encompassing drying, trimming, curing, and potentially advanced extraction and product manufacturing, is absolutely critical for determining the final quality, value, and utility of the cultivated material. While the MVP touches upon the basics, the full vision for Project Chimera includes a deep and nuanced simulation of these processes, offering players new avenues for skill expression, economic diversification, and product differentiation. (Doc1, Sec VII.E).

* **MVP Recap (as per Part 4, Sec 4.3.1):**
  + **Drying:** Manual placement on basic drying racks. Drying time is fixed or very simply influenced by rudimentary room humidity. Minimal quality impact.
  + **Curing:** Manual placement in basic curing containers (jars). Simple "burping" mechanic (e.g., daily button click). Minimal quality impact.
  + **Trimming:** Abstracted/simplified action to convert dried/cured material into a generic "sellable product." No detailed manual trimming.
  + **Exclusions:** No advanced extraction (concentrates), no edibles/topicals, no detailed environmental control for drying/curing, no quality tiers based on trimming.
* **Full Vision: Masterful Post-Harvest Techniques & Value-Added Product Creation:**

**5.8.1. Advanced Drying & Curing Environment Control (Full Detail):**Proper drying and curing are art forms that significantly impact the final quality (aroma, flavor, smoothness, potency preservation) of cannabis flower. The full vision simulates this with depth.

* + **Dedicated Drying Room/Zone:**
    - Players must designate or build a dedicated, well-sealed "Drying Room" or zone with its own environmental controls (separate from grow rooms).
    - **Environmental Parameters to Control:**
      * **Temperature:** Ideal range typically 18-21°C (65-70°F). Too warm = rapid drying, terpene loss. Too cold = very slow drying, higher mold risk.
      * **Relative Humidity (RH):** Ideal range typically 45-55% RH. Too high = mold risk, slow drying. Too low = too rapid drying, harsh product, terpene loss.
      * **Airflow:** Gentle, indirect airflow is crucial to prevent mold and ensure even drying. Stagnant air is detrimental. Too much direct airflow can over-dry buds quickly. Requires careful placement of OscillatingFanSOs or integration with the HVAC system (low CFM exchange).
      * **Light:** Drying rooms must be dark, as light degrades cannabinoids (especially THC to CBN) and terpenes.
    - **Equipment:** DehumidifierSO, HumidifierSO (if needed), SmallHeaterSO, SmallACUnitSO, OscillatingFanSO, networked TemperatureSensorSO & HumiditySensorSO. These are controlled manually or via PLCs/Central Computer with specific drying recipes.
    - **Drying Methods & Equipment:**
      * **Hang Drying (Whole Plant or Branches):** Player hangs harvested plants or large branches upside down from DryingLineSO or HangingRackSO items. Promotes slower, even drying.
      * **Net/Screen Drying (Individual Buds):** Buds are removed from branches (pre-trim or post-dry "bucking") and placed on multi-tiered MeshDryingRackSO items. Can dry faster, requires more handling.
  + **Drying Process Simulation:**
    - **Moisture Content Tracking:** Harvested plant material starts with high moisture content (e.g., ~75-80%). The drying process simulates the gradual reduction of this moisture. The target is typically 10-12% for curing, or slightly higher (12-15%) for some extraction methods.
    - **Drying Rate:** Influenced by:
      * Drying room temperature, RH, and airflow.
      * Bud size and density (denser, larger buds dry slower).
      * Initial moisture content.
    - **Quality Impact:**
      * **Too Fast Drying:** Harsh smoke/vapor, loss of volatile terpenes (aroma/flavor), chlorophyll not fully degraded (grassy taste).
      * **Too Slow Drying:** Increased risk of mold (Botrytis), potential for hay-like smell if not managed.
      * **Optimal Drying ("Low and Slow"):** Preserves terpenes, cannabinoids, results in smoother product. Takes longer (e.g., 7-14 in-game days).
    - **Monitoring:** Players use hygrometers in the drying room. An advanced (researched) WoodMoistureMeterSO tool (adapted for cannabis) could allow players to take direct readings of bud moisture content.
  + **Advanced Curing Process Simulation:**
    - **Curing Containers:** GlassJarSO (various sizes, amber option for UV protection), potentially CVaultSO (stainless steel, airtight) or GroveBagSO (specialized terpene-preserving bags - as high-tier consumables).
    - **Environment for Curing Jars:** Jars are ideally stored in a cool, dark, stable environment (a designated "Curing Zone" or within the controlled dry room).
    - **Internal Jar RH Management:** The crucial aspect.
      * Dried buds (at ~10-12% moisture) are placed in airtight containers. Moisture from the center of the buds slowly rehydrates the drier outer layers, raising the RH inside the jar.
      * **"Burping":** Player must regularly open the jars (e.g., once or twice daily for the first week, then less frequently) for 5-15 minutes to release excess moisture and exchange air. A JarBurpingScheduleReminder system in the UI helps track this.
      * **Target Jar RH:** Typically 58-62% RH for optimal long-term curing. Players use small MiniHygrometerSO items placed inside jars (or a tool that can measure jar RH when opened).
      * **Humidity Control Packs (Consumable):** TwoWayHumidityPackSO (e.g., simulating Boveda/Integra Boost) can be placed in jars to help maintain a target RH. Different RH % packs available.
    - **Curing Duration:** Weeks to months. Longer cures generally result in smoother product with more complex, refined aromas/flavors as chlorophyll breaks down and terpenes mature.
    - **Quality Impact of Curing:**
      * **Proper Cure:** Enhanced aroma, flavor, smoothness. Potential (subtle) perceived change in psychoactive effects due to cannabinoid/terpene interplay. Better preservation.
      * **Improper Cure (Too Wet/Infrequent Burping):** Mold risk, ammonia smell.
      * **Improper Cure (Too Dry):** Harsh, loss of aroma/flavor.
  + **C# Implementation (Full Vision for Drying/Curing):**
    - DryingRoomController.cs (linked to RoomEnvironmentController): Manages the specific environmental targets for drying.
    - HarvestedPlantBatch.cs: Represents a batch of harvested material, tracking its current moisture content, drying method, and accumulated quality modifiers from the drying process.
    - CuringJar.cs: Represents a jar of curing buds, tracking its internal RH (simulated based on bud moisture and burping), burp schedule, age, and accumulated quality modifiers from curing.
    - PostHarvestQualityModule.cs: Calculates final quality scores based on drying/curing parameters (duration, temp/RH stability, moisture content achieved).
    - UI for managing drying room conditions, setting burping reminders, and tracking curing jar status.
  + **AI Tooling (Full Vision for Drying/Curing):**
    - Cursor AI: Logic for moisture content simulation, CuringJar state management, quality modifier calculations.
    - AI for Concept Art: Visualizing ideal drying room setups or different curing containers.

Mastering drying and curing becomes a significant skill, allowing players to transform even an average harvest into a premium product, or ruin a great harvest with poor post-processing.

**5.8.2. Detailed Trimming Mechanics & Quality Tiers (Full Detail):** Trimming (manicuring) cannabis buds – removing excess sugar leaves and fan leaves – significantly impacts the product's appearance ("bag appeal"), smoothness, and market value.

* + **Wet Trim vs. Dry Trim:**
    - **Wet Trim:** Trimming buds immediately after harvest, before drying. Often easier as leaves are turgid. Can lead to faster drying.
    - **Dry Trim:** Trimming buds after they are fully dried (but before final curing). Often preferred for preserving terpenes and achieving a slower, more controlled dry. More delicate as trichomes can be brittle.
    - **Player Choice:** Player can choose their preferred method. Each might have slight GxE-like effects on final terpene retention or drying speed, or affect the efficiency/difficulty of the trimming process itself.
  + **Trimming Process & Tools:**
    - **Dedicated "Trimming Station" Equipment:** A TrimmingWorkstationSO (workbench with good lighting, comfortable chair - conceptual).
    - **Tools:** TrimScissorsSO (various types: spring-loaded micro-tip, curved blade, non-stick coated – offering slight differences in speed, precision, or "trichome disturbance"). TrimBinSO (a tray to collect fallen trichomes/kief).
    - **Manual Trimming Minigame (Optional, High Engagement):**
      * A focused minigame where the player uses mouse controls to "snip" away leaves from a 3D bud model.
      * Scored on precision (avoiding cutting into the bud itself), speed, and thoroughness.
      * Higher scores result in better quality trim ("Hand-Trimmed Premium" tier) and more kief collected.
      * Lower scores or rushed jobs result in "Roughly Trimmed" or "Machine-Trimmed Equivalent" quality.
    - **Skill-Based Abstracted Trimming (Alternative to Minigame):**
      * Trimming is a timed action. The player's "Trimming Skill" (unlocked/leveled via Skill Tree or practice) and the quality of their TrimScissorsSO influence:
        + Speed of trimming (grams per hour).
        + Quality outcome (probability of achieving different trim tiers).
        + Amount of kief recovered.
  + **Trim Quality Tiers & Market Value:**
    - **Untrimmed/Rough:** Lowest value, may not be accepted by some buyers/contracts.
    - **Machine Trimmed (Simulated):** If player uses a late-game AutomatedTrimmerSO machine. Fast, but lower quality, more trichome loss.
    - **Standard Hand Trim:** Decent quality, good market value.
    - **Premium Hand Trim / Connoisseur Trim:** Meticulously trimmed by hand (high score in minigame or high skill level). Highest bag appeal, commands premium price.
    - **Kief Collection:** Trimming (especially dry trimming over a TrimBinSO) yields kief (loose trichomes) as a separate valuable byproduct, which can be sold or used to make hash/rosin.
  + **C# Implementation (Full Vision for Trimming):**
    - TrimmingController.cs: Manages the trimming process (minigame logic or skill-based timed action).
    - TrimQualityCalculator.cs: Determines the final trim tier and kief yield based on player performance/skill, tools, and trim method.
    - HarvestedBudBatch.cs: Stores the trim quality tier as part of its data.
    - Inventory system to track kief as a separate item.
  + **AI Tooling (Full Vision for Trimming):**
    - Cursor AI: Logic for the skill-based trimming calculations, minigame scoring (if implemented).
    - AI for 3D Assets: Models for different trim scissors, trim bins. Visuals for differently trimmed buds (less/more sugar leaf).

Detailed trimming adds another layer of skill expression and economic optimization, allowing players to add significant value to their harvested product.

**5.8.3. Advanced Extraction Techniques (Post-MVP - Full Detail):** The creation of cannabis concentrates (hash, rosin, oils, shatter, wax, isolates) is a major post-MVP expansion, introducing new product lines, complex equipment, and scientific processes. (Doc1, Sec VII.E).

* + **Solventless Extraction Methods:**
    - **Dry Sift / Kief Tumbling:**
      * **Equipment:** KiefTumblerSO / SiftingScreenSetSO (various micron sizes).
      * **Process:** Agitating dried cannabis flower or trim over screens to separate trichome heads (kief). Different screen sizes yield different grades of kief.
      * **Output:** Kief (various grades). Can be pressed into hash.
    - **Ice Water Extraction (Bubble Hash):**
      * **Equipment:** IceWaterExtractionKitSO (multiple micron-graded filter bags - "bubble bags"), IceMakerSO, MixingVesselSO (food-grade bucket), StirringToolSO (manual or electric).
      * **Process:** Cannabis material is agitated in ice-cold water. Trichomes become brittle and break off, then filtered through bags of decreasing micron size to separate different grades of hash.
      * **Output:** Bubble Hash (various grades, e.g., "Full Melt 6-star," "5-star," "Food Grade"). Quality depends on starting material, technique, and water temperature.
    - **Rosin Pressing:**
      * **Equipment:** RosinPressSO (various sizes, pressure ratings, plate materials - e.g., pneumatic, hydraulic). RosinBagSO (filter bags of various micron sizes). ParchmentPaperSO.
      * **Process:** Applying heat and pressure to cannabis flower, kief, or hash (placed in rosin bags) to squeeze out rosin oil.
      * **Parameters:** Temperature, pressure, duration of press are critical and player-controlled (or recipe-driven). Different parameters yield different results (e.g., "Live Rosin" from fresh-frozen material, "Flower Rosin," "Hash Rosin").
      * **Output:** Rosin (various consistencies: shatter-like, budder, sauce). Yield and quality depend heavily on starting material quality and press technique.
  + Solvent-Based Extraction Methods (Abstracted for Gameplay - Doc1, Sec VII.E):  
    These involve flammable/volatile solvents and complex safety procedures. Gameplay will abstract the core chemical processes while focusing on equipment operation, parameter control, and safety management.
    - **Light Hydrocarbon Extraction (e.g., Butane - BHO, Propane - PHO):**
      * **Equipment:** ClosedLoopExtractionSystemSO (extraction vessel, solvent tank, recovery pump, collection chamber), VacuumOvenSO (for purging residual solvents), DewaxingColumnSO (for winterization).
      * **Process (Abstracted):**
        1. Material packed into extraction column.
        2. Chilled solvent washes over material, dissolving cannabinoids and terpenes.
        3. Solvent is recovered (evaporated and re-condensed) leaving crude oil.
        4. Crude oil is purged of residual solvent in a vacuum oven (critical step for safety/quality).
        5. Optional Winterization/Dewaxing: Removing fats, lipids, waxes by dissolving crude oil in ethanol, freezing, then filtering to improve clarity/purity.
      * **Output:** BHO/PHO concentrates like Shatter, Wax, Budder, Live Resin (if fresh-frozen material is used).
    - **Ethanol Extraction:**
      * **Equipment:** EthanolSoakingVesselSO, RotaryEvaporatorSO (for solvent recovery), FiltrationSystemSO.
      * **Process:** Soaking cannabis material in cold ethanol. Ethanol is then evaporated to leave crude oil. Often used for full-spectrum extracts or as a precursor to distillation.
      * **Output:** Crude ethanol extract (FECO - Full Extract Cannabis Oil).
    - **Supercritical CO2 Extraction:**
      * **Equipment:** SupercriticalCO2ExtractorSO (very expensive, high-tech).
      * **Process:** Using CO2 at high pressure and temperature (supercritical fluid state) as a tunable solvent to selectively extract cannabinoids and terpenes.
      * **Output:** High-purity CO2 oil, often used for vape cartridges or further refinement (distillate).
  + **Post-Extraction Refinement:**
    - **Distillation:**
      * **Equipment:** ShortPathDistillationKitSO or WipedFilmEvaporatorSO.
      * **Process:** Further refining crude extracts to isolate individual cannabinoids (e.g., THC distillate, CBD distillate) by heating under vacuum.
      * **Output:** High-potency (90%+) clear distillate. Terpenes are often stripped during this process and can be re-added later.
    - **Crystallization (Isolate Production):**
      * **Equipment:** ReactionVesselSO, CentrifugeSO (lab-scale), ChromatographySystemSO (very advanced).
      * **Process:** Further purifying distillates or extracts to produce crystalline isolates of single cannabinoids (e.g., THCA crystals, CBD isolate powder - 99%+ pure).
      * **Output:** THCA/CBDA crystalline, CBD isolate.
  + **Quality Factors for Extracts:**
    - **Potency:** Cannabinoid concentration.
    - **Purity:** Absence of residual solvents, lipids, waxes, contaminants.
    - **Terpene Profile:** Richness and complexity of aroma/flavor (especially for "live" extracts).
    - **Consistency/Texture:** (Shatter, budder, sauce, crystalline).
    - **Clarity/Color.**
  + **Safety Systems (for Solvent Extraction):**
    - Requires a dedicated, well-ventilated "Extraction Lab" zone with ExplosionProofFanSO, GasDetectorSO (alerts for solvent leaks).
    - Failure to manage safety (e.g., improper purging, leaks) can lead to simulated accidents (fires, explosions - game over for that lab or severe financial penalty/facility damage).
  + **C# Implementation (Full Vision for Extraction):**
    - ExtractionProcess.cs (base class) with derived classes for each method (e.g., RosinPressProcess.cs, BHOClosedLoopProcess.cs). Each manages its own state machine, equipment requirements, input material, processing parameters, and output products/yields/qualities.
    - ExtractionLabController.cs: Manages all active extraction processes, safety systems.
    - Equipment\_Extractor.cs (base class for all extraction equipment) with specific logic for each type.
    - Complex UI for controlling extraction parameters, monitoring progress, and viewing lab safety status.
    - Product\_ConcentrateSO.cs: Defines different types of concentrates and their properties.
  + **AI Tooling (Full Vision for Extraction):**
    - Cursor AI: Logic for the state machines of different extraction processes, parameter effect calculations, safety system logic.
    - AI for Concept Art: Visualizing the complex extraction equipment and lab layouts.

Advanced extraction transforms Project Chimera into a chemical engineering and product development simulation, offering high-risk, high-reward gameplay for advanced players.

**5.8.4. Edibles & Topicals Manufacturing (Post-MVP - Full Detail):** Creating infused products like edibles and topicals is another major post-MVP expansion, building upon the availability of extracts or decarboxylated flower. (Doc1, Sec VII.E).

* + **Infusion Basics:**
    - **Decarboxylation:** Heating cannabis flower or extracts (e.g., THCA -> THC, CBDA -> CBD) to activate cannabinoids for oral/topical use. Requires an OvenSO or DecarboxylationReactorSO with precise temperature/time control.
    - **Lipid Infusion:** Infusing decarboxylated cannabinoids into fats/oils (butter, coconut oil, MCT oil) as these are good carriers. Requires InfusionVesselSO (e.g., heated mixing tank, double boiler).
  + **Edibles Manufacturing:**
    - **Product Types:** Gummies, chocolates, baked goods (brownies, cookies), tinctures, capsules. Each represented by an EdibleProductSO.
    - **Recipe-Based System:**
      * EdibleRecipeSO: Defines ingredients (e.g., infused oil/butter, flour, sugar, flavoring, gelatin), quantities, mixing/cooking/baking steps, target dosage per serving.
      * Players unlock recipes via research or purchase.
    - **Equipment:** IndustrialMixerSO, DepositorSO (for gummies/chocolates into molds), BakingOvenSO (commercial scale), EncapsulationMachineSO, TinctureBottlingLineSO.
    - **Dosage Consistency & Quality Control (CRITICAL):**
      * The primary challenge in edibles is achieving accurate and consistent dosage (e.g., mg of THC per gummy).
      * The game will simulate variability based on:
        + Accuracy of infusion process.
        + Precision of ingredient measurement by player (or automated equipment).
        + Homogeneity of mixing.
      * **In-Game Lab Testing (Consumable EdibleTestKitSO or NPC service):** Players can (and should) test batches of edibles for dosage consistency.
      * **Consequences of Inaccuracy:**
        + Products with wildly inconsistent dosages (too high or too low) will get poor market reviews, reputation penalties, or even simulated "product recalls" (financial loss).
        + Selling accurately dosed, consistent edibles builds strong brand reputation.
  + **Topicals Manufacturing:**
    - **Product Types:** Creams, balms, salves, lotions. Each an TopicalProductSO.
    - **Ingredients:** Infused oils, waxes (beeswax, carnauba), butters (shea, cocoa), essential oils (for scent/additional effects – links to terpene system?).
    - **Equipment:** CosmeticMixingVesselSO (emulsifiers, homogenizers), FillingMachineSO (for tubes/jars).
    - **Quality Factors:** Consistency, texture, scent, cannabinoid concentration.
  + **Packaging & Labeling:**
    - **Equipment:** PackagingLineSO (automated or semi-automated).
    - **Consumables:** PackagingMaterialSO (jars, bottles, bags, boxes).
    - **Labeling Requirements (Simulated):** Products need labels displaying strain (if applicable), cannabinoid content (THC/CBD mg per serving/package), ingredients, batch number, warnings. An in-game "Label Designer" UI might allow players to create compliant labels. Incorrect labeling leads to penalties.
  + **C# Implementation (Full Vision for Edibles/Topicals):**
    - InfusionProcess.cs, DecarboxylationProcess.cs.
    - EdibleManufacturingLine.cs / TopicalManufacturingLine.cs: Manages a sequence of processing steps using different equipment.
    - RecipeSystemManager.cs: Handles EdibleRecipeSO and TopicalRecipeSO.
    - DosageCalculator.cs: Calculates theoretical dosage based on inputs.
    - QualityControlModule\_InfusedProducts.cs: Simulates dosage variability and links to lab testing results.
    - LabelingManager.cs: Checks product labels against simulated regulatory requirements.
  + **AI Tooling (Full Vision for Edibles/Topicals):**
    - Cursor AI: Logic for recipe execution, dosage calculations, quality control simulations, labeling checks.
    - AI for Text Generation: Assisting with creative product descriptions, ingredient lists for labels.
    - AI for Image Generation: Designing packaging concepts, product mockups.

Edibles and topicals manufacturing adds product diversification and a new layer of quality control challenges, appealing to players interested in creating finished consumer goods.

**5.8.5. Quality Control, Lab Testing, and Product Labeling (Integrated System):** This system formalizes the process of assessing product quality and ensuring compliance, becoming increasingly important as players create more advanced and regulated products.

* + **In-House Lab vs. Third-Party NPC Lab Services:**
    - **Early Game:** Players rely on basic observation, their own senses (abstracted "aroma/flavor check" action), and simple tools.
    - **Mid-Game (Research Unlock):** Can build a basic "In-House QC Lab" module with equipment like:
      * BenchtopHPLC\_SimulatorSO (High-Performance Liquid Chromatography): For testing cannabinoid potency (THC, CBD, CBG percentages). Requires SolventSO consumables and ReferenceStandardSOs for calibration. Has a running cost and sample processing time. Provides a CannabinoidProfileReport.
      * GasChromatography\_SimulatorSO (GC-MS): For testing terpene profiles (percentages of key terpenes). Similar consumable/calibration needs. Provides a TerpeneProfileReport.
      * ResidualSolventTesterSO: For checking purity of solvent-based extracts.
      * MicrobialTestingKitSO: For checking for mold/bacteria contamination.
    - **Late-Game / Alternative:** Send samples to a more accurate (but more expensive and time-consuming) third-party NPC "Certified Lab Service." This might be required for official "Certificates of Analysis" (CoAs) needed for high-value sales or specific contracts.
  + **Product Grades & Certificates of Analysis (CoAs):**
    - Based on lab test results (potency, purity, terpenes, contaminants, residual solvents) and other quality factors (trim quality, cure quality, dosage consistency for edibles), products are assigned a grade (e.g., A, B, C, Fail, or more nuanced like "Connoisseur Grade," "Mid-Shelf," "Budget," "Not For Sale").
    - CoAs (digital in-game items) are generated, summarizing test results. These can be attached to product batches and influence market value and buyer trust.
  + **Batch Tracking & Traceability:**
    - Each production run (harvest batch, extraction batch, edibles batch) gets a unique Batch ID.
    - Lab tests are linked to Batch IDs.
    - This is crucial for (simulated) product recalls if a batch fails QC or causes negative feedback.
  + **Regulatory Compliance (Abstracted):**
    - As players scale and sell more widely (especially in the post-MVP player-driven market), they may need to adhere to simulated regulatory requirements for testing, labeling, and product safety. Failure results in fines, reputation loss, or market access restrictions.
  + **C# Implementation:**
    - QualityControlManager.cs: Oversees lab testing requests, result generation, product grading.
    - LabEquipmentController.cs (for HPLC, GC, etc.): Simulates test process, consumable use, calibration needs.
    - CertificateOfAnalysis.cs: Data object for storing test results.
    - BatchTracker.cs: Assigns and manages Batch IDs.
    - RegulatoryComplianceModule.cs: Checks products against current simulated regulations.
  + **AI Tooling:**
    - Cursor AI: Logic for simulating lab test processes, generating CoAs, grading algorithms.

This QC system adds a layer of professionalism and realism, rewarding meticulous players and creating consequences for cutting corners.

**5.8.6. Specialized Equipment for Advanced Processing (Summary):** The full vision requires a significant expansion of placeable equipment, each represented by an EquipmentDataSO with unique properties, costs, power/utility needs, operational parameters, and research prerequisites. This includes (but is not limited to):

* + **Drying/Curing:** Advanced environmental control units for dry rooms, specialized curing containers, humidity control pack dispensers.
  + **Trimming:** High-end trim scissors, automated trimming machines.
  + **Solventless Extraction:** Various sized/spec Kief Tumblers, Ice Water Extraction setups (washers, multiple bag sets), Rosin Presses (manual, pneumatic, hydraulic with different plate sizes/materials).
  + **Solvent-Based Extraction:** Closed-Loop BHO/PHO Extractors (multiple sizes/capacities), Vacuum Ovens (various sizes), Rotary Evaporators, Winterization Freezers/Filtration, Supercritical CO2 Extractors.
  + **Refinement:** Short Path Distillation Kits, Wiped Film Evaporators, Crystallization Reactors, Lab-Scale Centrifuges, Chromatography Systems.
  + **Edibles/Topicals:** Decarboxylation Ovens/Reactors, Infusion Vessels, Industrial Mixers, Depositors, Commercial Baking Ovens, Encapsulation Machines, Homogenizers, Filling/Bottling Lines.
  + **Lab/QC:** HPLC & GC machines, Residual Solvent Testers, Microbial Testers, Digital Microscopes, Lab Benches, Fume Hoods, Glassware.
  + **Packaging:** Automated/Semi-Automated Packaging Lines, Label Printers.

Each piece of equipment will have a 3D model, UI for interaction/control, and specific effects on the processing simulation. Their acquisition and effective use are key to unlocking advanced product lines.

By implementing this full vision for Post-Harvest Processing, Project Chimera will offer players a comprehensive "seed-to-sale" (and beyond) experience, where cultivation mastery is just the beginning. The ability to transform raw flower into a diverse range of high-value, quality-controlled products will provide immense strategic depth and long-term economic gameplay.