Introduction

Hey fellow devs,

Thank you to everyone who has supported this project, tested systems, sent feedback, or built your own worlds with it. Seeing what you create and how you use these tools in ways I didn't expect is what keeps me building.

When we released V3, it already felt like a massive step. Since then, I honestly never stopped building on top of it.

V4 didn't start as "let's make a new version."
It started as: this can be cleaner… this can scale better… this edge case should never happen again… this system deserves a stronger backbone.

Over time, that mindset turned into thousands of commits, deep refactors, structural cleanups, new systems, production validation with professional studios, and a lot of invisible work that only shows its value when you push things to the limit.

- Large World Partition maps.
- ense foliage + full day/night + weather.
- Nested inventories.
- Multiplayer replication.
- AI perception loops.
- Factory-scale logic.

V4 is the result of pushing everything further, without losing what made V3 powerful.

Thank you for building with this ecosystem and pushing it forward with your feedback.

With warmth,
Eric Ruts
In the name of the whole GamesByHyper team.

Contents

What V4 Is

V4 is the most complete modular gameplay ecosystem currently available for Unreal Engine.

It consists of **100+ interoperable systems** designed to work together seamlessly while remaining fully modular and independently extensible. Each system can operate on its own, but together they form a cohesive production foundation.

This is not a genre template. It is a **cross-genre framework** capable of supporting:

- Oblivion-style RPG progression
- ARK-style survival systems
- GTA-style open worlds
- Rust-like PvP
- The Forest / Valheim-style co-op survival
- Satisfactory / StarRupture-style factory automation
- Subnautica-style exploration
- Competitive multiplayer frameworks
- Large-scale single-player narrative worlds
- Single player, Co-op, PvP, Open World Multiplayer, Roguelike, and hybrid combinations

There is no other publicly available Unreal ecosystem that combines **this scale of systems, this level of integration, and this architectural cohesion** into a single framework.

V4 represents **tens of thousands of development hours** and years of iteration.

The engineering investment behind this framework represents **millions of dollars in development value**.

But beyond the numbers, this is personal.

As a child, I always dreamed of building worlds. Not just one game, but a foundation capable of powering any kind of game. RPGs, survival worlds, factory simulations, competitive multiplayer, open-world adventures. A system not locked into one genre or one production.

Originally, this framework was being built as **internal tooling**. A production backbone for larger projects. At some point, that changed. Instead of keeping it closed, I chose to make it available.

Today, that foundation is accessible to:

- **Hobby developers**
- **Indie studios**
- **Professional teams**
- **Larger studios exploring modular production workflows**

V4 is not a template collection. It is the realization of that vision.

It is a scalable production framework built to power serious games.

V4 Rollout & Versioning

Here is how V4 will roll out.

- V3 will remain available on **Unreal 5.5 and 5.6** as a stable base.
- V2 on the versions before
- V4 is built for **Unreal 5.7 and higher** and is the path forward.

I'm rolling this out in phases on purpose.

Phase 1

First, I will publish V4 versions of:

- **MST Pro**
- **Online Multiplayer Framework**
- **Environment Building Toolkit**

Even with heavy internal testing, once thousands of developers start using something, new edge cases appear. That's normal.

If you find issues::

- Send it.
- Be specific.
- I will fix them quickly.

That way everything runs on the same architecture before anything new gets added.

Important

If you are in active production:

- Do not migrate immediately from V3 to V4
- Do not mix V3 and V4 systems
- Wait until the full V4 ecosystem is released and confirmed stable
  - Based on past experience. Over 95% of all bugs are reported and fixed within the first month of release.

V4 systems are designed to work together. Mixing versions will cause conflicts.

Step 2 – Full Migration

Once the core systems are stable, all remaining existing systems will move to V4.

Step 3 – Expansion

After the full ecosystem is stable, new V4 systems will release in waves.

Strong foundation first.

Then expansion.

Changelog

Architecture and Foundations

Event Manager (tag-based orchestration)

V4 introduces a unified **Event Manager** system.

Instead of wiring systems directly together, you can now:

- Register listeners to a gameplay tag
- Send events with structured payloads
- Cascade logic across unrelated systems

Mental model

- Sender
- Tag
- Payload
- Listener

Example

- You define a tag like: Eventtrigger.LevelInstance.Load
- A Quest, Dialogue, or Console Command can send it with payload (level asset, location, parameters)
- Anything listening to that tag executes

Same idea works for:

- Add item
- Send private message
- Trigger sequence
- Start machine logic
- Fire UI prompts

This keeps systems modular and makes complex flows data-driven.

Container-Based Inventory (major foundation rewrite)

The inventory system has been rebuilt.

Inventory is no longer “slots only”.
It is **container-based**.

Containers can:

- Contain slots
- Contain other containers
- Exist in-world
- Be opened, dropped, nested, transferred
- Inject logic (magazines, equipment containers, machine inputs)

MST Pro example > Two backpack types:

- Backpack A: increases capacity
- Backpack B: right-click → open container → drag items → drop on ground

Weapons can accept magazine containers automatically.

This unlocks:

- Inventory inside inventory
- Future machine crafting pipelines
- Factory compatibility
- Real modular loadouts

Large World + Performance Direction

V4 was validated on large World Partition environments with:

- Dense foliage
- Full dynamic weather
- Full day-night cycle
- Large Multiplayer sessions

Key performance direction work includes:

- Object pooling for hot paths (combat hits, impacts, UI spam patterns)
- Save validation strategies (avoid saving unchanged/no-op values)
- Replication stability passes
- Initialization improvements for streamed environments
- Reduced unnecessary ticking and UI churn

Project-Wide Structure Improvements

Naming and component placement clarity

Project-wide consistency improvements:

- Components that belong on PlayerController, Gamestate or GameMode use clear naming (example: **AC_PC_*, AC_GS_*** and **AC_GM_*)**
- GameState assets clearly marked (example: GS_*)
- Character assets clearly marked (example: CH_*)
- Structs standardized (F_*)
- Enums standardized (E_*)

The goal is simple:

- You can tell what something is and where it belongs instantly.

UI and UX Improvements

Complete UI overhaul

The UI has been completely revamped.

- Cleaner visual style
- More professional presentation
- Unified design language
- Modular reusable widgets

This impacts the feel of every template and system out of the box.

Generic Radial Menu (new reusable UI primitive)

A generic **Radial Menu** framework was created so multiple systems can reuse the same foundation instead of shipping hard-coded one-offs.

Used by:

- Building
- Emotes
- Interaction-style menus
- Future systems that need fast radial selection

Core System Enhancements (existing systems upgraded)

This section is for **existing systems** that were upgraded/refactored in V4.

Ability System + Attribute Manager

- Attribute Manager refactored with gameplay tags
- Better base class structure
- More robust state effects
- Contagious state effects
- Cleaner integration between abilities and attributes

Damage logic and combat correctness

Damage routing now prioritizes:

- Object-based validation
- Team Affiliation checks
- Attribute-based rules

Instead of relying on simplistic "enemy/damageable" tagging, damage is decided by who you are, who they are, and what rules apply.

Combat Framework

- Updated to align with container inventory
- Cleaner hit processing and response handling
- Better integration with pooling patterns for repeated effects
- Multiplayer correctness improvements

Equipment Manager

- Rebased and improved
- Better integration with container inventory and combat
- Improved equip/unequip handling and stability

AutoLandscape Material

- Performance improvements for Nanite workflows
- Custom paint layering support
- Better landscape/foliage integration direction

Background Music System

- Better aligned with tags and audio workflow
- Improved handling via audio submix organization

Building System

- Reworked base classes
- Cleaner gameplay tech foundation
- Replaced hard-coded menu logic with the generic Radial Menu foundation
- Improved structure for integration with inventory and data flows

Cave Building Toolkit

- No major changes, compatibility maintained

Chat System

- Updated and aligned with modernized architecture patterns
- Improved reliability and usability

Console Command Manager

- Updated and improved
- Cleaner integration patterns for triggering gameplay and debug logic

Day-Night Cycle, Weather System, Time Manager

- Performance improvements
- Visual upgrades
- Better integration with environment workflows
- Large-map validation outcomes

Environment Building Toolkit

- Upgraded to align with the improved environment pipeline
- Better integration with day/night + weather direction

Farming System

- UI improvements
- Performance improvements
- General usability refinements

Fishing

- Simplified setup via drag-and-drop components
- Easier assignment to volumes and environments

Foodstuff System

- Minimal changes

Global Save System

- Reported issues resolved over time
- Improved performance validation
- Save filtering patterns (avoid saving unchanged/no-op values)
- Safer behavior in complex streamed environments

Icon Creator

- More linked with building/world content workflows
- Better usability for icons in more systems

Information Prompt System

- Generic queued prompt system (objectives + notifications + general info)
- Better reusability across modules

Interaction System

- Better feasibility checks:
  - line of sight blocking
  - more accurate radius validation
- Clearer interactable feedback

Interactive Foliage

- Improved stability and integration direction for large environments

Inventory, Crafting, Vendor, Loot Chests, Hotbar

- Inventory completely migrated to containers
- Crafting revamped and optimized
- Recipe logic expanded (tool-quality modifiers like can opening/spillage)
- Vendors and loot containers updated to align with container flows
- Hotbar stability improvements

Inventory Jigsaw

- Improved drag/drop positioning
- Better usability and stability under container system

List-Based Inventory

- Updated to align with containers

Map System

- Major improvements including Region integration support
- Tier-based capture system for large maps:
  - high-level view uses lower density textures
  - deeper zoom levels switch to grid-based higher detail tiles
- Much better quality on large worlds without the usual performance cost

Menu and Gamepad System

- Upgraded usability
- New UI
- Better flow and navigation feel

Mesh-to-Actor Swap System

Completely revamped:

- New save/load technique
- Better behavior in World Partition regarding ISMS
- Supports PCG workflows more cleanly
- Handles add/remove rules and streamed world realities much better
- Cleaned up uncencecarry variables

Mineable Rocks

- Damage improvements (actor health component)
- Chaos fixes and stability improvements

Online Multiplayer Framework

- Upgraded stability and ease of use
- Better single-player + multiplayer flow handling

Outline System

- Minor improvements / compatibility

Quest Manager

- Major upgrades
- Stronger modular trigger patterns
- Better compatibility with event-driven workflows

Replication Subsystem

- Quality-of-life improvements
- Better stability across systems

Respawn

- Improved stability and flows for common multiplayer patterns

Skill Level Manager

- Improved usability
- Better UI and refined logic

Swimming System

- Quality-of-life improvements and polish e.g. bubbles

Tree Cutting

- Chaos bug finally fixed
- Ease-of-use improvements (stick spawning, stability, polish)

Visual Novel

- Upgraded with video support
- First-play/startup options
- More flexible presentation

Voice Manager

Fully revamped:

- Voice selection is character-driven (on the pawn)
- Multi-character support:
  - multiple voices stored
  - system selects best match (e.g., masculine/feminine variants)
  - fallback behavior if needed
- Much easier to scale voice content cleanly in multiplayer character setups

New Systems Introduced in V4

These systems were created or formalized during the V4 cycle. (New modules will be released separately unless noted. Most of the systems shown below are near ready to be released)

Foundational systems are already integrated across the V4 ecosystem and are also available as separate modules.

Core Infrastructure

These systems form part of the architectural backbone of V4 and are already integrated across multiple modules.

- **Event Manager**
  Tag-based event routing system that allows you to register listeners and dispatch structured payloads across systems.
  Usaly used by Questing, Dialogue, Building, Level Instances, Console Commands, and more.
  Enables actions such as spawning levels, adding items, triggering sequences, or sending private messages without hard wiring systems together.
  Also available as a standalone module.
- **Actor Health Component**
  Self-contained health component that manages damage, state effects, and health logic.
  Integrated into trees, rocks, buildings system etc. Also usable independently.
- **Unified Tag Mapper**
  Centralized mapping system that links gameplay tags to structured data (items, buildables, recipes, reputation, etc.).
  Automatically retrieves structured information based on tags.
  Used across inventory, crafting, building, and progression systems.
  Also available as a separate module.
- **Object Pooler**
  Performance-focused pooling system for repeated effects such as damage indicators and combat responses.
  Reduces spawn/despawn overhead and improves performance under load.
  Integrated into Combat and Attribute systems.
- **Player Proximity Manager**
  Enables or disables actors and logic based on player distance.
  Used in AI, Fishing, world interaction systems, and large maps to reduce unnecessary processing.

- **Generic Radial Menu**
  Reusable radial UI framework e.g. used by Building, Emotes, and interaction-based selection systems.
  Modular and extendable.
- **Information Prompt System (Queued Prompt Manager)**
  Centralized queued notification system that manages objectives, warnings, contextual hints, and system feedback.
  Prevents UI spam and ensures structured player messaging across systems.
  Integrated into multiple gameplay modules.

AI & World Systems

- **NPC Framework**
  StateTree-based AI framework supporting routine-driven NPC behavior, advanced perception, combat logic and team-aware reactions.
  Designed for scalable world populations and systemic AI behavior for large RPG's.
  Don't confuse this with a companion which can do personal actions like chopping actual trees.
- **Region Manager**
  Define world regions by drawing borders and triggering logic when entering or leaving regions.
  Useful for faction zones, biome control, or rule-based world sections.
- **Permission System**
  Manage team-level and global permissions for access control such as doors, systems, or restricted areas.
- **Team Affiliation System**
  Governs hostility, damage validation, and alignment checks.
  Used across Combat, Ability, and AI logic.
- **Reputation Manager**
  Track NPC and faction reputation values that influence dialogue, access, and world responses.
- **Memory Manager**
  Previously integrated internally, now also available as a standalone module on request.
  Allows storing, retrieving, and modifying persistent memory states.

Gameplay Systems

- **Companion System**
  Taming and command system allowing players to train, feed, and assign actions to companions. Current setup is tranq the AI, feed it and tame. Including companion storage like Pokemon, Palworld etc.
  Supports follow, attack, and task-based behaviors.
- **Mount System**
  Framework for mounting and dismounting actors with extensible behavior and animation hooks.
- **Guild Manager**
  Online-capable guild system with permission control, guild levels, and structured management.
  Supports progression and organized group play.
- **Guide System**
  Context-aware guidance system for tutorial-style prompts, warnings, and dynamic player feedback.
- **Inspection System**
  Allows players to inspect items and objects in detail.
  Supports 3D previews, audio playback, document reading, and integration with Databank and Guide systems. Also includes voice integrations.
- **Buried Storage**
  Concealed storage system using physical material checks for hidden containers in the world.
- **Lootable Corpse**
  Allows direct interaction with corpses instead of spawning generic loot bags.
- **Corpse Decomposition**
  Timed decay system with visual effects and environmental feedback such as particles and progression states.
- **Skinning System**
  Resource harvesting system triggered after kills with animation and reward handling.
- **Data Bank System**
  In-game documentation system allowing storage of tutorials, images, videos, and structured information within a UI tab.
- **3D Character Previewer**
  Drag-and-drop component that renders a live 3D character preview inside UI menus.

Combat & Defense

- **Defense System Primitive**
  Trap-based and dungeon-style defensive mechanics.
- **Defense System Modern**
  Advanced automated defenses including turrets, hitscan weapons, flamethrowers, and homing rockets.
- **Security System**
  Modular security infrastructure including CCTV cameras, lasers, gates, and sirens. Fully interactive and extensible for controlled areas or base defense.
- **Weapon Attachment System**
  Modular weapon loadout system allowing visual customization and attribute-based stat modification.

Streaming & Sequencing

- **Level Instance Manager**
  Dynamically load and unload level instances for modular world building and narrative events.
- **Sequence Manager**
  Centralized control for playing and skipping cinematic sequences.

Factory Systems

- **Conveyor Systems** – Item transport pipelines
- **Machine Systems** – Automated production and processing units
- **Power Management** – Energy generation and distribution logic
- **Battery Systems** – Stored energy handling and grid balancing
- **Network Distribution** – Resource and signal routing logic
- **Resource Extraction** – Automated extraction systems
- **Production Chains** – Structured multi-step crafting pipelines

Fully integrated with container inventory, permissions, multiplayer replication, and event-driven architecture.

Im so extremly proud of whats to come and what we have. Thank you so much for being with me on this journey.

What you build on top of it is what will truly define it.

Thank you again for being part of this journey.

Eric Ruts