عالیه—ایده، دامنه و گفت‌وگوهایی که با AI داشتی رو خوندم. کانسپتت (ترکیب ONI + DST با سناریوی سفر در زمان و حلقه‌های محیطی لایه‌به‌لایه) پتانسیل بالایی داره. راهنمایی AI یونیتی خوب بود، اما برای یک توسعه‌دهنده‌ی تنها لازمِ شُسته‌رفته‌تر و اجرایی‌تر بشه تا از پراکندگی جلوگیری کنه. این هم بازخورد و یک روپ‌مپ عملیاتی که بتونی همین الان باهاش جلو بری:

### خلاصه‌ی بازخورد

* ✅ مسیر **«Loop اصلی → محیط/اکسپلور → NPC → Tech Tree → روایت → آرت → پولیش»** منطقیه.
* ⚠️ اما برای **Solo-dev** باید به شکل **Vertical Slice‌های کوچک با معیار پذیرش مشخص** شکسته بشه؛ وگرنه ریسک پراکندگی بالاست.
* 🎯 تمرکز اولیه باید روی **حلقه‌ی بازی شفاف و قابل‌تست** باشه؛ **Tech Tree** و **روایت** را به اسلایس دوم منتقل کن.
* 🧭 چون آرت نقطه‌ضعفته، از همون ابتدا یک Pipeline آرت حداقلی (Low-Poly استایل ثابت + ایزومتریک ثابت) تعریف کن که دست‌وپاتو نبنده.

### ستون فقرات طراحی (Core Loop پیشنهادی)

> Gather → Craft → Equip/Build → Unlock Area (Ring بعدی) → Survive

این حلقه باید در یک بیومِ پایه (Ring-1) ظرف ۱۰–۱۵ دقیقه تجربه شود.

### Vertical Slices (دو فاز اول کاملاً اجرایی)

#### ✅ Vertical Slice 1 (MVP قابل بازی – ۲ هفته)

هدف: یک راند ۱۰–۱۵ دقیقه‌ای که بازیکن در Ring-1 زنده بماند، آیتم بسازد، یک ساختمان بسازد، و به آستانه‌ی Ring-2 برسد.

محدوده

* **گیم‌پلی:**
  + ۱ بیوم (Ring-1) با ۱ خطر محیطی + ۱ دشمن ساده.
  + ۳ آیتم پایه (Resource, Food, Tool) + ۱ Modifier کاربردی (مثلاً افزایش Energy).
  + ۳ رسپی Crafting + ۱ ساختمان تولیدی (Workbench یا Firepit).
  + سیستم Hunger/Energy/Health فعلی با بالانس حداقلی.
* **UI:**
  + Inventory UI پایدار (استکینگ، استفاده، Equip).
  + Crafting UI بر اساس Category با نمایش Requires/Available.
* **دوربین ایزومتریک:**
  + زاویه‌ی ثابت (Pitch ~35–45°، Yaw ثابت)، Orthographic برای خوانایی، اندازه‌ی ثابت.
* **کنترل:**
  + Input System جدید + Pick-up اتوماتیک (که داری).
* **دیتابیس/ادیتور:**
  + تولید محتوا از طریق ادیتور اختصاصی‌ات (که مزیت کلیدی توست).

معیار پذیرش (Acceptance)

* از صفر تا ساخت اولین ساختمان و زنده‌ماندن ۱۰ دقیقه بدون کرش.
* TTF-Craft (زمان تا اولین Craft) < ۵ دقیقه.
* حلقه‌ی Gather→Craft→Equip واضح و بدون گم‌گشتگی.

#### 🚀 Vertical Slice 2 (Progression کم‌ریسک – ۲ تا ۳ هفته)

هدف: عمق دادن به حلقه با Progression سبک بدون پیاده‌سازی کامل Tech Tree.

محدوده

* **NPCهای اولیه:**
  + ۱ NPC قابل جذب (Idle → Follow → Simple Task: Gather یا Guard).
  + سیستم ساده‌ی وفاداری/انگیزه (مثلاً یک متغیر واحد که با غذا/امنیت بالا می‌رود).
* **Progression سبک (Pre-TechTree):**
  + «Tier Token» یا «Permit» که با انجام یک مینی‌هدف آزاد می‌شود و موجزاً ۲–۳ رسپی جدید را فعال کند.
* **اکسپلوریشن حلقه‌ای:**
  + Ring-2 با نیازمندی مشخص (مثلاً لباس خاص یا آیتم ضدسم)؛ اگر بازیکن شرط را ندارد، جریمه/مرگ منطقی.
* **UI بازخورد پیشرفت:**
  + Badge/Toast برای “New Recipes Unlocked” + مقصد بعدی.

معیار پذیرش

* بازیکن با کمک NPC ساده، یک Permit بگیرد و رسپی‌های Tier-2 فعال شوند.
* Ring-2 ورودی واضح، شرط واضح، و بازخورد شکست/موفقیت داشته باشد.

### ترتیب اجرای پیشنهادی (Sprintهای یک‌هفته‌ای)

* **هفته ۱ — Core Loop پایدار**
  + قفل کردن ایزومتریک (زاویه/فاصله/OrthographicSize ثابت).
  + تکمیل Inventory (استکینگ، استفاده، Equip رخدادمحور).
  + ۳ آیتم + ۳ رسپی + ۱ ساختمان + دشمن/خطر پایه.
  + بالانس اولیه‌ی Drain (پروفایل‌های Normal/Hard برای تست).
* **هفته ۲ — MVP آماده‌ی تست**
  + Polish Crafting UI (Disabled states، Tooltip نیازمندی).
  + پیام‌های رویداد (OnCraft/OnConsume/OnDamage → UI).
  + بازی‌پذیری ۱۰–۱۵ دقیقه + تست کاربر محدود.
* **هفته ۳ — NPC ساده + Gate به Ring-2**
  + NPC: Follow/Task بسیار ساده + یک منبع داده در دیتابیس.
  + Gate منطقی به Ring-2 (شرط تجهیز/آیتم).
  + Token/Permit برای آزادسازی رسپی‌های Tier-2.
* **هفته ۴ — Feedback & Balancing**
  + Telemetry سبک (Log داخلی):
    - Time-to-First-Craft, Death Cause, Resource Scarcity.
  + بالانس مجدد Drain/Drop/Costs.
  + جمع‌بندی Slice 2 و تصمیم برای Tech Tree واقعی.

### تعریف «آماده برای توسعه / آماده برای تحویل»

* **Definition of Ready (DoR):**
  + هر فیچر یک سناریوی بازی‌پذیر دارد، Data آن در SO ثبت شده، UI Mock (اگر لازم است) مشخص است.
* **Definition of Done (DoD):**
  + Scenario پاس می‌شود، بدون ارور/GC Spike محسوس، Logged metrics ثبت می‌شود، حداقل ۱ تست دستی مستند.

### ریسک‌ها و راه‌حل‌ها

* **پراکندگی دامنه:** فقط فیچرهایی را وارد Sprint کن که به خروجی قابل بازی هفته کمک می‌کنند.
* **فشار آرت:** از یک Palette/Kit Low-Poly ثابت استفاده کن و Style Guide کوتاه بنویس (Scale، رنگ‌های مجاز، مقدار Outline/Gradient=0).
* **ایزومتریک‌سازی سخت:** به Orthographic پایبند بمان؛ زاویه/نور را ثابت نگه دار تا آرت ساده‌تر دیده شود.
* بالانس بقامحور: با Telemetry ساده (Log) تصمیم بگیر، نه حدس.

### KPIهای سبک برای هدایت طراحی

* TTF-Craft < ۵ دقیقه
* Survival-10min Rate (نرخ بقا تا ۱۰ دقیقه) ≥ ۶۰٪ در Ring-1
* Time-to-Gate به Ring-2 ≈ ۱۲–۱۸ دقیقه
* ۸۰٪ از مرگ‌ها باید از ۲ علت واضح بیایند (گرسنگی/سم) → یعنی بازخوردها روشن‌اند.

### بُرد کاری سبک (برای مدیریت شخصی)

ستون‌ها: Backlog → Next → Doing → Verify (DoD) → Done → Icebox

برچسب‌ها: core-loop, ui, db, content, npc, balance, bug.

### پیشنهادهای کوچک اما اثرگذار

* **Content Budget** برای هر اسلایس بنویس: دقیقاً **«۳ آیتم + ۳ رسپی + ۱ ساختمان + ۱ دشمن»**.
* **Prefab/Scriptable Templates** بساز تا ورود محتوای جدید ۱–۲ دقیقه طول بکشد.
* **Event-Driven UI** (که پایه‌اش را داری) را معیار کن؛ تمام سیستم‌ها از Eventها تغذیه کنند.
* Fail-fast: اگر در Ring-2 شرط بقا نداری، خیلی سریع و با پیام واضح شکست بده (Reader-Friendly).

اگر بخوای، همین برنامه را به صورت یک چک‌لیست Sprint هفتگی یا Kanban Markdown هم درمی‌آورم تا مستقیم بذاری توی پروژه/لینکدین.

می‌خوای من همین رو برات تبدیل به یک چک‌لیست Sprint هفتگی Markdown کنم که راحت بذاری داخل پروژه/لینکدین؟