0) What we are actually “downloading”

Repos you will clone

1. **Cosmos Cookbook** (recipes + docs + referenced scripts)
2. **Cosmos Reason2** (examples/utilities + install scaffolding; inference can run without cloning, but cloning is useful for the provided scripts)

Recipes we will use (inference patterns)

- **Egocentric Social & Physical Reasoning (Reason2 inference)** as the *core reasoning engine pattern*
  NVIDIA Recipes Guide
- **Video Search & Summarization (Re
  NVIDIA Recipes Guide
  the *continuous video analytics pattern* (we adapt the pattern; we don’t build search)
  NVIDIA Recipes Guide

What we explicitly do NOT downl

NVIDIA Recipes Guide

- Post-training recipes (we keep inference-only; this is a design decision and is defensible)

---

1) Clone Cosmos Cookbook (recipes + docs)

This repo is heavy on media; **Git LFS is required**. The repo quick start also standardizes `uv` + `just`.

```
# 1) Install Git LFS
sudo apt update && sudo apt install -y git-lfs
git lfs install

# 2) Install uv + just (as recommended by the cookbook)
curl -LsSf https://astral.sh/uv/install.sh | sh
source $HOME/.local/bin/env
uv tool install -U rust-just
```

```
# 3) Clone cookbook repo
git clone https://github.com/nvidia-cosmos/cosmos-cookbook.git
cd cosmos-cookbook

# 4) Install cookbook deps
just install

# 5) (Optional) Serve docs locally for easy browsing
just serve-external
```

Why this matters to your project:

- The cookbook is where you pull **prompt patterns and recipe conventions** from.
- It cleanly separates `docs/` vs `scripts/` which helps you reference recipes while keeping your own repo minimal.

---

2) Install Cosmos Reason2 (two valid paths)

Option A — Clone `cosmos-reason2` and run their examples (recommended)

NVIDIA's Reason2 installation guide explicitly supports **virtualenv via `uv`** or Docker.

```
# system deps
sudo apt-get update
sudo apt-get install -y curl ffmpeg git git-lfs unzip

# clone reason2
git clone https://github.com/nvidia-cosmos/cosmos-reason2.git
cd cosmos-reason2

# install uv (if you didn't already)
curl -LsSf https://astral.sh/uv/install.sh | sh
source $HOME/.local/bin/env

# huggingface auth (needed to pull model weights)
uv tool install -U huggingface_hub
```

```
hf auth login

# install python deps (CUDA extra depends on your driver/CUDA)
uv sync --extra cu128
source .venv/bin/activate

# run their minimal sample script
python scripts/inference_sample.py
```

This matches both the **official installation page** and the **repo README** structure.

Option B — Minimal inference without cloning the repo

Reason2 is included in **transformers>=4.57.0**, and NVIDIA's reference page points to the minimal sample invocation.

If you're moving fast, Option A is still better because you get the ready-made scripts + prompt YAMLs.

---

3) Choose model size based on your GPU

The Reason2 repo specifies minimum GPU memory:

- **2B**: 24GB
- **8B**: 32GB

So:

- If you're on a 24GB GPU: start with **2B**.
- If you're on 32GB+: you can use **8B**, but for this competition you usually don't need it.

---

4) (Optional) Start an inference server (useful for your agent loop)

NVIDIA's Reason2 reference recommends **vLLM** for serving and provides an example command.

This is optional, but it can make your "video_agent.py" calls clean.

---

5) Pull the recipe patterns you need (without dragging in extra scope)

Your project's "NVIDIA Recipes Guide" file already locks the mapping:

- **Egocentric Social & Physical Reasoning** = core reasoning engine
- **Video Search & Summarization** = pattern for continuous video querying
- **Physical Plausibility** = inspiration for constraint sanity checks (not training)
- **Intelligent Transportation post-training** = explicitly not used

Concrete workflow to "download/use recipes" without bloat:

1. **Clone cookbook** (done in step 1).
2. Use cookbook **docs locally** to copy:
   - prompt structure conventions
   - example "what to ask the model"
   - any structured-output patterns you want
3. In your own repo, implement **only**:
   - `reasoning/cosmos_reasoner.py` wrapper (calls Reason2)
   - prompt templates tuned to your hazard taxonomy + outputs

That stays aligned with your plan: **reasoning-first integration, no training**.

---

6) Wire into your project repo in the smallest possible increments

Step 6.1 — Create a "smoke test" script first

Before you build the full agent loop, create a single script that:

- loads 1 short clip (5–10s per demo blueprint)
  DEMO SCENARIOS — RAINBOW SIX SI…
- runs Reason2 once (or per N frames)
- prints:
  - hazards found
  - drone safety
  - human-follow safety
  - recommendation
  - explanation

This matches your expected demo outputs and avoids early architectural mistakes.

Step 6.2 — Only after smoke test passes, add:

- constraint gating (human vs drone affordances)
- DEMO SCENARIOS — RAINBOW SIX SI…
  arness (Expected vs Observed)
  Project Plan NVIDIA Cosmos

---

7) Checklist for "done with downloads / ready to build"

You're "done with setup" when:

- `cosmos-cookbook` clones successfully (Git LFS ok) and `just serve-external` runs
- `cosmos-reason2` installs and `python scripts/inference_sample.py` runs
- You can run one short indoor clip through Reason2 and get a stable textual
  Project Plan NVIDIA Cosmos
  r hazard taxonomy