Tier■1 Sandbox (DeFi) — Steps 2, 3, 4

Goal: Run DeFi MVP with synthetic latents using matched■filter + Stage■10 parser only, while keeping Stage■11/WDD and ARC paths intact for Tier■2.

Step 2 — Disable WDD/Denoiser in MVP

Set rails to bypass Warp→Detect→Denoise. Keep the parser path active.

Policy JSON (sandbox profile):

```
{ "rails": { "denoise": false, "use_wdd": false }, "ltv_max": 0.75, "mapper": { "model_path":
".artifacts/defi_mapper.joblib", "confidence_threshold": 0.7 }, "oracle": { "age_sec": 5,
"max_age_sec": 30 } }
```

CLI example:

```
python3 milestones/defi_milestone2.py --rails stage11 --runs 5 --policy "$(cat
configs/tier1_mvp.json)" --context '{"oracle":{"age_sec":5,"max_age_sec":30}}'
```

Step 3 — Encapsulate Configs (No Code Changes)

Introduce a small config switch to route around Stage■11. Prefer environment variable for quick toggling.

Environment variable:

```
SANDBOX=1 # enables matched■filter+parser■only mode
```

Optional policy merge override:

```
POLICY_OVERRIDES='{"rails":{"denoise":false,"use_wdd":false}}'
```

Makefile target (excerpt):

```
mvp_all: python3 milestones/defi_milestone1.py --rails stage11 --policy "$$(cat
configs/tier1_mvp.json)" python3 milestones/defi_milestone2.py --rails stage11 --runs 5
--policy "$$(cat configs/tier1_mvp.json)" python3 milestones/defi_milestone4.py --rails
stage11 --runs 5 --policy "$$(cat configs/tier1_mvp.json)" python3
milestones/defi_milestone5.py --rails stage11 --runs 5 --policy "$$(cat
configs/tier1_mvp.json)"
```

Step 4 — Preserve Tier■2 Hooks (No■Ops in Sandbox)

Keep the Stage■11 functions present but inert when SANDBOX=1.

Pseudocode shim (rails module):

```
def warp(latents, policy): if os.getenv("SANDBOX") == "1" or policy.get("rails",
{}).get("use_wdd") is False: return latents # no■op return warp_impl(latents, policy) def
detect(traces, policy): if os.getenv("SANDBOX") == "1" or policy.get("rails",
{}).get("use_wdd") is False: return stock_matched_filter(traces) # Stage■10 path return
detect_impl(traces, policy) # Stage■11 def denoise(state, policy): if os.getenv("SANDBOX") ==
"1" or policy.get("rails", {}).get("denoise") is False: return state # no■op return
denoise_impl(state, policy)
```

Contract to keep stable:

- Stock parser API unchanged: returns (keep, order, confidence).

- Verifier interface returns {'ok': bool, 'reason': str} across all paths.
- Aux hooks (aux.stage10/aux.stage11) optional; sandbox mode may omit them without failing tests.

Smoke Tests to Validate Sandbox

1. M1: Non■exec prompt abstains (empty plan or verify.ok=False with reason='abstain_non_exec').
2. M2: Five repeated runs yield identical top■1 (denoise off).
3. M4: Blocked actions (withdraw_high_ltv, borrow_low_hf) set verify.ok=False with reasons containing 'ltv'/'hf'.
4. M5: Consolidated suite passes with policy rails.denoise=false, rails.use_wdd=false.