**So let's access the current state of our repository, to determine the paths and the viability of these changes to our coding and an exact plan to implement these changes and the appropriate next steps:**

**✅ PRIMARY REFACTOR PLAN (Snapshot-First Logging Architecture)**

**🧠 THE CORE PROBLEM**

**❌ Legacy Behavior Was Overcomplicated:**

* **Market confirmation was checked in both should\_log\_bet() and again in write\_to\_csv()  
  → Causing valid bets to be skipped after passing all filters**
* **Pending logic was split across:**
  + **pending\_bets.json**
  + **micro\_topups.json**
  + **market\_conf\_tracker.json**
  + **theme\_exposure\_tracker.json**
* **Simulations re-ran every 30 minutes, but monitor\_early\_bets.py used stale stake & EV% from when the bet was first queued**
* **Spread and h2h exposures were tracked separately, even though they’re logically the same bet**

**✅ THE REFACTOR PLAN**

**We move to a snapshot-first architecture. This means:**

**Snapshots become the only source of truth for determining whether a bet is eligible to log, needs confirmation, or qualifies for a top-up.**

**🔧 THE NEW SNAPSHOT ROW STRUCTURE**

**Each row in snapshot.json contains everything needed:**

**{**

**"game\_id": "2025-06-23-ARI@CWS-T1940",**

**"market": "h2h",**

**"side": "ARI",**

**"segment": "1st\_5",**

**"theme\_key": "ARI\_h2h",**

**"normalized\_theme\_key": "ARI\_sides",**

**"theme\_segment\_key": "ARI\_sides::1st\_5",**

**"sim\_prob": 0.5281,**

**"market\_prob": 0.5452,**

**"ev\_percent": 4.92,**

**"adjusted\_kelly": 1.62,**

**"logged": false,**

**"queued": true,**

**"baseline\_consensus\_prob": 0.5271,**

**"snapshot\_stake": 0.0,**

**"last\_logged\_time": null**

**}**

**📌 Key Additions Explained:**

| **Field** | **Meaning** |
| --- | --- |
| **queued** | **This bet is pending confirmation (was skipped earlier)** |
| **baseline\_consensus\_prob** | **Market prob at time the bet was first skipped (fixed reference)** |
| **normalized\_theme\_key** | **E.g. "ARI\_h2h" and "ARI\_spread" → "ARI\_sides"** |
| **theme\_segment\_key** | **Combo key used for exposure tracking** |
| **adjusted\_kelly** | **Current stake suggested based on fresh sim/market odds** |
| **snapshot\_stake** | **Optional tracking of stake suggested in this specific snapshot** |
| **last\_logged\_time** | **(Optional) timestamp of last log for auditing/duplication check** |

**🧮 MARKET CONFIRMATION: CENTRALIZED**

**All market confirmation logic lives in should\_log\_bet(), using this logic:**

**movement = current\_market\_prob - baseline\_consensus\_prob**

**required = required\_market\_move(hours\_to\_game, num\_books, ev\_percent)**

**if movement >= required:**

**✅ Confirm the bet**

**else:**

**❌ Still queued**

**✅ No market confirmation logic exists in write\_to\_csv() anymore.**

**🧪 STAKE LOGGING DECISION LOGIC**

**🔁 Exposure Tracking Strategy (Option B)**

**We do not store exposure in snapshots. Instead:**

* **Load existing exposure from market\_evals.csv with load\_existing\_stakes()**
* **Track additional exposure within the current loop using session\_exposure**

**🔁 Exposure Key Normalization**

**We treat h2h and spread as the same exposure pool:**

**python**

**CopyEdit**

**def normalize\_theme\_key(theme\_key: str) -> str:**

**if "\_h2h" in theme\_key or "\_spread" in theme\_key:**

**return theme\_key.replace("\_h2h", "\_sides").replace("\_spread", "\_sides")**

**return theme\_key**

**theme\_segment\_key = f"{normalize\_theme\_key(theme\_key)}::{segment}"**

**🧠 Exposure-Based Logging Rules**

**python**

**CopyEdit**

**exposure\_from\_csv = existing\_exposures.get(theme\_segment\_key, 0.0)**

**exposure\_from\_loop = session\_exposure.get(theme\_segment\_key, 0.0)**

**total\_exposure = exposure\_from\_csv + exposure\_from\_loop**

**stake = snapshot\_row["adjusted\_kelly"]**

**delta = stake - total\_exposure**

**if total\_exposure == 0 and stake >= 1.0:**

**✅ First bet → log full stake**

**elif total\_exposure > 0 and delta >= 0.5:**

**✅ Top-up → log delta**

**else:**

**❌ Skip → too small or redundant**

**✅ CSV Logging Format**

**Your market\_evals.csv will include:**

**csv**

**CopyEdit**

**date, game\_id, market, side, segment, stake, ev\_percent,**

**sim\_prob, market\_prob, baseline\_consensus\_prob,**

**movement, required\_movement, entry\_type, source\_snapshot**

**✅ Every logged bet carries:**

* **Why it was confirmed (movement)**
* **Where it came from (snapshot)**
* **What type it was (first/top-up)**

**🔁 LOOP FLOW OVER SNAPSHOT ROWS**

**In monitor\_early\_bets.py or log\_betting\_evals.py:**

**python**

**CopyEdit**

**existing\_exposure = load\_existing\_stakes("market\_evals.csv")**

**session\_exposure = defaultdict(float)**

**for row in snapshot:**

**if not row["queued"]:**

**continue**

**if not MIN\_EV <= row["ev\_percent"] <= MAX\_EV:**

**continue**

**theme\_segment\_key = f"{normalize\_theme\_key(row['theme\_key'])}::{row['segment']}"**

**total\_exposure = existing\_exposure[theme\_segment\_key] + session\_exposure[theme\_segment\_key]**

**stake = row["adjusted\_kelly"]**

**delta = stake - total\_exposure**

**if total\_exposure == 0 and stake >= 1.0:**

**✅ log\_bet(row, stake)**

**session\_exposure[theme\_segment\_key] += stake**

**elif total\_exposure > 0 and delta >= 0.5:**

**✅ log\_bet(row, delta)**

**session\_exposure[theme\_segment\_key] += delta**

**🧼 Removed/Deprecated Files**

| **File** | **Replaced By** |
| --- | --- |
| **pending\_bets.json** | **Replaced by "queued": true in snapshot row** |
| **market\_conf\_tracker.json** | **Snapshot baseline\_consensus\_prob** |
| **theme\_exposure\_tracker.json** | **CSV + in-loop tracking via session\_exposure** |
| **micro\_topups.json** | **❌ Removed — logic handled by exposure delta checks** |

**✅ FINAL CHECKLIST**

| **Feature** | **Status** |
| --- | --- |
| **Snapshot is single source of truth** | **✅** |
| **Market confirmation logic is centralized in should\_log\_bet()** | **✅** |
| **Stale stake logic removed from monitor\_early\_bets.py** | **✅** |
| **Theme+segment exposure normalized via \_sides** | **✅** |
| **Logging flow avoids duplicate logs via loop memory** | **✅** |
| **CSV includes full movement + confirmation trace** | **✅** |
| **All bet types (first/top-up) handled without extra queues** | **✅** |

**✅ NEXT STEP OPTIONS**

**Would you like me to:**

1. **🔧 Rewrite should\_log\_bet() to accept a snapshot row and confirm bets with movement check**
2. **🔧 Rewrite monitor\_early\_bets.py to loop over a snapshot file and apply this logic**
3. **🛠 Create a utility to normalize exposure keys and load exposure totals**
4. **✨ Bundle all of the above into a full working module (snapshot\_logger\_v2.py)**

**Let me know what you want done first — or if you want a full implementation drop.**

**✅ Updated Interpretation of Your Notes (Snapshot-First Context)**

**🧠 Original Problem (Micro Top-Ups):**

• Top-ups < 0.5u were getting stored in micro\_topups.json separately, creating complexity.

**✅ Updated View:**

* You’ve replaced pending\_bets.json, micro\_topups.json, etc., with **just snapshot rows**.
* Instead of queuing micro top-ups, we now **check delta vs exposure in the loop using snapshot + CSV**.

**🔧 Fix:**

* ✅ **Top-ups with Δ stake ≥ 0.5u are allowed**
* ❌ Top-ups with Δ < 0.5u are **skipped**, **but not stored** anywhere else.

**✅ Result:**

* No more micro\_topup queue
* Cleaner logic
* Same protection against overlogging
* ✅ ✅ This still holds and should be applied exactly as written, just **within snapshot rows and your logging loop**.

**🔄 Optional Cleanup: Quiet Hours Notification Queue**

• Some bets are logged but not announced due to quiet hours queueing

**✅ Still true, but now:**

* If you're using snapshot-based logging and a Discord alert system, this can either:
  + Be **tightly coupled with log\_bet()** (i.e., alert when logged), or
  + Be **delayed until write\_to\_csv() confirms the log** and is outside quiet hours

🔧 **Suggestion**:  
If quiet hours only affect **notifications**, **keep it isolated** to Discord logic, **don’t wrap it into logging logic**. This way:

* write\_to\_csv() always logs
* Discord alerts just suppress for now, queue if needed

✅ This remains optional, but yes — it still applies.

**✳️ Updated Summary of Actions (Snapshot-First Version)**

| **Area** | **Updated Action** |
| --- | --- |
| ✅ Confirmation Logic | ✅ Fully handled in should\_log\_bet() using snapshot row’s baseline\_consensus\_prob and latest market\_prob |
| ✅ Pending Tracking | ❌ Delete pending\_bets.json, instead track "queued": true inside snapshot row |
| ✅ Exposure Tracking | ✅ Continue scanning market\_evals.csv via load\_existing\_stakes() + session\_exposure |
| ✅ Micro Topups | ✅ Skip if Δ < 0.5u — do not store them. **No micro\_topups.json needed** |
| 🔄 Quiet Hour Queue | ✅ Still valid — keep queueing of alerts (not logs) if you want silence overnight |

**💥 Benefits (Still Fully Valid)**

| **Benefit** | **Applies?** | **Notes** |
| --- | --- | --- |
| ✅ More bets logged | Yes | Especially top-ups that previously got stuck in micro queues |
| ✅ Simpler logic | Yes | Single snapshot row, no multiple trackers |
| ✅ Fewer limbo bets | Yes | All queued logic is in snapshot rows only |
| ✅ Easier debugging | Yes | Full log eligibility logic is consolidated in one spot |

**✅ Final Summary**

Yes — **everything you originally wrote is still correct** — we just:

* Drop pending\_bets.json and micro\_topups.json
* Track "queued" and baseline\_consensus\_prob in the **snapshot row itself**
* Use CSV + session memory to handle exposure
* Keep notification suppression logic **separate** from logging decisions

Would you like me to start with:

1. ✅ Writing the new should\_log\_bet(snapshot\_row, exposure\_tracker) logic
2. ✅ Rewriting monitor\_early\_bets.py to process snapshot rows using this system
3. ✅ Full simplified visual flowchart of the entire refactored system (to document or share)

Let me know what format you want next.