Revenue Management — README (Checkpoint)

> **Collaboration mode:** one tiny Codex prompt per change; Codex saves a **new file each time** (e.g., `Step 34 – description.html`). This README reflects the latest confirmed state.

---

1) System Overview

- **Purpose:** Rules-based, explainable pricing for multifamily. No market comps, no competitor tracking.
- **Scope:** New-lease pricing + Renewals from a single uploaded rent roll.
- **Design stance:** Internal, "walled-garden" logic (defensible, no antitrust signaling).

---

2) Data Ingestion

- **Input:** CSV/XLS rent roll with, at minimum: `UnitID`, `Floorplan`, `Status`, `CurrentRent`, `LeaseEnd` (optional but recommended: `PreleaseStart`).
- **Mapping:** UI confirm step binds columns → internal fields. After mapping, **Box Score & charts render immediately**.

---

3) Occupancy Math (Authoritative)

- **Occupied % (snapshot):** `occupied / total`, where *occupied-like* = status starts with "occupied" **or** contains "notice".
- **Projected % (trending):** `1 - ((Vacant + Notices - Prelease) / N)`, clamped to `[0,1]`.
- `isVacant = status includes "vacant"`
- `isNotice = status includes "notice"`
- `preleases = count(PreleaseStart truthy)` but **capped** at `(vacant + notices)`
- **No time window limits** (no 120-day horizon).
- **Used by:** Box Score **and** all pricing logic via `computeTrending()`.

---

4) Single-Threshold Trend Control (the only occupancy knob)

- **Control:** `Comfort Target Trend %` (T), e.g., `95%`.
- **Deadband:** ±0.5pp with small-sample widening `+ 0.7pp/√n`.
- **Community gate:**
- **GateDown:** `T−2pp` → block increases when community ≤ GateDown.
- **GateUp:** `T+1pp` → mild boost to increases when community ≥ GateUp.
- **Distance ramp:** Movement scales with `|FP − T|` (full step at ~5pp).
- **Step sizes:** gentle `0.75%`, standard `1.5%`, fast `2.5%`.

- **Weekly decrease cap:** negative moves limited by existing `maxWeeklyDec`.
- Implemented in `computeMoveSingleThreshold(cfg, fpTrend, commTrend, nFp)`.

---

5) New-Lease Pricing Flow (current)

1. **Trending:** `tState = computeTrending(norm)` (community + per-FP) using Box Score math.
2. **Move direction:** For each floorplan, `dir = computeMoveSingleThreshold(...)`.
3. **Base:** `adjustedBase = fpAvg * (1 + dir)` (respect weekly dec cap).
4. **Per-term price:** Apply term premium (`shortTermAdj`) and month multiplier (`seasonalityMultiplier`).
5. **Explainability:**
6. **Inline Notes column (same row):** `Term premium X% & over cap (N) Y% = Z%` (Z% is net vs adjusted base).
7. **Debug line A:** Reference term (longest) arrow `↑ / ↓ / →`, price, and `move%`.
8. **Debug line B:** `FP vs Target ±pp` and **Community gate** state (Blocked / Neutral / Boost).

---

6) Renewals Flow (current)

Anchor rule (baseline, no premiums): - **Baseline new** for the floorplan (no term premiums, no month pressure): `baselineNew = fpAvg * (1 + dir)`. - **Percent-to-New:** Offer moves from **Current → Baseline** by `cfg.pctToNew` (e.g., 50%). - If `Current < Baseline`: `offer = curr + pctToNew * (baseline - curr)`. - If `Current ≥ Baseline`: `offer = allowDecAbove ? curr - pctToNew * (curr - baseline) : curr`. - **Guardrails (max-only):** If enabled, cap **increases only** to a max% (no mins; do not cap decreases). Uses UI fields `Max % (below new)` / `Max % (above new)`; whichever applies. - **Explainability:** - **Inline Notes (right cell):** Anchoring and applied % are shown. - **Footer debug:** `Current $C vs baseline-new $B • pct-to-new: X% • max: Y% → applied ±Z%`.

---

7) UI Controls (relevant today)

- **Strategy → Single threshold:** `Comfort Target Trend %` (replaces Low/High; Low/High are hidden fallbacks that auto-sync for backward compatibility and history restore).
- **Pricing Adjustment Style:** gentle / standard / fast (sets base step size).
- **New-Lease Terms:** selectable range (e.g., 2–14 months). Reference term (for debug) is the **longest** selected.
- **Seasonality Curve:** chosen curve for month multipliers.
- **Renewals:**
- **Percent-to-New (%):** e.g., 50%.
- **Allow Renewal Decrease:** on/off.
- **Apply Guardrails to All Terms:** on/off (currently used for **max-only** cap on increases).
- **Max % (below new)** / **Max % (above new):** applied only as max increase caps when guardrails are enabled. *Min% fields are presently ignored by design.*
- **Renewal Terms:** selectable range (e.g., 2–14 months).

8) Files & Steps (recent)

- `Step 22.html` — Baseline before single-threshold work.
- `Step 23 – add comfort target.html` — Added Target input (no logic changes).
- `Step 24 – wire comfort target cfg.html` — Config + history snapshot/restore for target.
- `Step 25 – add single threshold logic.html` — Added `computeMoveSingleThreshold()` (not wired).
- `Step 26 – wire single-threshold into New Pricing.html` — New-lease pricing uses single-threshold.
- `Step 27 – show target logic debug line.html` — Debug line under NL.
- `Step 28 – trending equals box score.html` — Trending identical to Box Score.
- `Step 29 – debug shows reference term direction.html` — Arrow + reference term debug.
- `Step 30 – add target gate debug line.html` — Added Target/Gate debug.
- `Step 31 – inline term math at right.html` — Added per-term Notes column (NL).
- `Step 32 – remove low-high controls.html` — Removed Low/High UI; hidden fallbacks synced from Target.
- `Step 33 – wire single-threshold into Renewals.html` — Renewals use single-threshold move.
- `Step 34 – renewals anchored to baseline no-premiums.html` — Renewals offers = pct-to-baseline (no premiums) + **max-only** increase cap.

---

9) Operator-Facing "Why" (chips & notes)

- **New Pricing:**
- Chip 1: *your existing text* (kept), plus debug lines showing reference term direction and Target/Gate.
- Notes per term: `Term premium X% & over cap (N) Y% = Z%`.
- **Renewals:**
- Notes per term: `Anchored to baseline (no premiums) • pct-to-new K% • max M% → applied ±Z%`.
- Footer: current vs baseline with applied % and knobs.

---

10) Test Playbook

1. **Target gate test:** Set Target=95%. Feed community 92% with a high-trend FP (≥96%). Expect **blocked increases** (dir=0). Raise community to 96% → increases allowed (boosted).
2. **Deadband jitter:** FP within ±0.5pp of Target should show `move≈0`. Reduce sample size—deadband widens automatically.
3. **New-lease vs renewals coherence:** For a FP with fpAvg=$1,500 and dir=+10% → `baselineNew=$1,650`. With pct-to-new=50%, renewals long terms ≈ $1,575 unless max%<+5%.
4. **Max-only cap:** Set max 3%; with curr=$1,500 and baseline=$1,650 → halfway $1,575 caps to **$1,545** (+3%).

5. **Decreases allowed:** Set allowDecAbove=On; when curr>$baseline, renewals should move down by pct-to-new and **ignore** max cap.
6. **Seasonality stress (NL only):** Confirm per-term notes add up to observed price deltas.

---

11) Known Gaps / Next Micro-Steps

- (A) **Expose gates & deadband as advanced settings** (optional UI reveal): GateDown/GateUp offsets, deadband base, small-sample widening factor.
- (B) **Reference term switcher:** Toggle reference for debug (Longest ↔ Best-priced).
- (C) **Renewals footer counts:** Optionally add *allowed vs actual expirations* for the reference month.
- (D) **Snapshot hygiene:** When loading very old snapshots (with only Low/High), we infer Target from midpoint—keep hidden fallbacks synced.
- (E) **QA harness:** Seedable demo dataset + scripted assertions for dir, baseline, caps.

---

12) Key Functions (signatures)

```
// trend parity with Box Score
function computeTrending(norm) -> { tComm, tFP, occPct }

// single-threshold move (community gate + distance ramp + sample dampening)
function computeMoveSingleThreshold(cfg, fpTrend, commTrend, nFp) -> dirFraction
```

---

13) Security & Compliance

- No competitor price ingestion; no market comps; single-property logic only.
- Deterministic rules; human-override via floors/ceilings and max-increase caps.

---

14) How we work (saved)

- One change per Codex prompt; always save a new file.
- Keep UI visible changes minimal; add debugs first; then swap logic.
- Maintain backward compatibility (hidden fallbacks; tolerant snapshot restores).

End of README (Checkpoint).