We already have global chat, DB, rate limits, guest mode, Permit2 foundations, and a `/api/verify/worldid` skeleton. We need to **make guest posting rock‑solid**, wire **real Cloud Verify v2** end-to-end, polish **landing/composer UX**, and **add Dark Mode** (toggle, system, sunrise→sunset). Work in *plain English first*, then change code only where necessary. Keep feature flags and environment-driven config.

🎯 Top objectives (do these in order)

1. **Guest Mode — fix the regression & unify limits**
   - Server policy for guests: **60 chars**, **10 messages/day**, **30s cooldown**.
   - Guests must be able to post ≤60 chars **without** the "World ID required" banner.
   - Only verified-only paths (e.g., >60 chars, star/report, Work Mode) should require verification.
2. **Real World ID verification (Cloud Verify v2)**
   - Implement **server‑side** proof validation against World Developer Cloud Verify v2.
     - Base: `https://developer.worldcoin.org` (Developer Portal API).
     - REST endpoint: `POST /api/v2/verify/{app_id}` with JSON body: `{ nullifier_hash, merkle_root, proof, verification_level, action, signal? }`.
     - Only trust proofs after **server** verification (never client-only).
     - If a `signal` is used, hash it appropriately server-side before verifying.
   - On success: upsert user → role=`verified`; store **SHA‑256 of nullifier_hash** (UNIQUE) and set `wm_uid` cookie; no PII.
   - On failure: return **400** with clear human reasons (`expired_proof`, `action_mismatch`, `duplicate_nullifier`, …).
3. **Landing & composer UX (friendlier + obvious verify)**
   - Replace **"Admin Access"** with primary **"Verify with World ID"** CTA (opens MiniKit Verify). UI updates to verified **without reload**.
   - Replace "Posting is human-only" with:
     "Guests can say hi (60 chars, 10/day). Verify with World ID to unlock full chat and features."
   - Composer badges: show **Guest Mode** + `60 char limit • 10/day • 30s cooldown`.
   - Map server reasons to friendly banners (see Acceptance).
4. **Diagnostics & observability**
   - `/api/policy` returns all limits.
   - `/api/me` returns `{ role, guestStats?, verifiedStats?, theme? }`.
   - Add `/api/worldid/diag` → `{ appId, action, apiBase, verificationLevel }` (no secrets).

   - Log reason codes for blocked messages: `guest_length_exceeded`, `guest_cooldown`, `guest_daily_limit`, `verified_only_action`, `content_filter_low_entropy`, etc.
5. **Content filter tweak (reduce false positives)**
   - Allow natural elongations (e.g., *Howdyyyyy*) and emojis. Only block if a **single character repeats >5** *and*message entropy is very low. Keep obvious spam checks.
6. **Future hooks (do not overbuild)**
   - Keep **Permit2** behind `ENABLE_PERMIT2=0`.
   - Leave stubs for "friends + DM" tables (`connections`, `dm_threads`, `dm_members`, `dm_messages`) with indexes/uniques, **no UI yet**.
   - Keep "emoji shop" behind `EMOJI_SHOP_ENABLED=0` (economy later, reusing Room Rain points).
7. 🌙 **Dark Mode (toggle • system • sunrise→sunset)**
   Add first-class theming with four modes:
   - **Light**
   - **Dark**
   - **System** (follows `prefers-color-scheme`)
   - **Auto (Sunrise → Sunset)**
8. **Implementation:**
   - **CSS tokens:** Introduce theme tokens via CSS variables.
     - File: `client/src/index.css` (or create `client/src/theme.css`).
     - Define defaults on `:root` for **light**; define overrides on `:root[data-theme="dark"]`.
     - Tokens to include at minimum: `--bg`, `--card`, `--panel`, `--text`, `--muted`, `--border`, `--accent`, `--success`, `--warning`, `--error`, and shadow/elevation vars.
     - Update components (banners, buttons, inputs, chat bubbles, tooltips, skeletons) to use tokens, not hard-coded colors.
   - **Theme state & provider:**
     - Create `client/src/hooks/use-theme.ts` and `client/src/theme/ThemeProvider.tsx`.
     - Persist **mode** in `localStorage("wm_theme_mode")` with values: `"light" | "dark" | "system" | "sun"`.
     - Compute **active theme** on mount:
       - If `mode==="light"` → apply light.
       - If `mode==="dark"` → apply dark.
       - If `mode==="system"` → use `matchMedia('(prefers-color-scheme: dark)')`.
       - If `mode==="sun"` → decide by local sunrise/sunset (see below).

        - Apply by setting `document.documentElement.dataset.theme = 'light' | 'dark'`.
        - Listen for `prefers-color-scheme` changes when `mode==="system"`.
    - **Sunrise→Sunset logic:**
        - Attempt `navigator.geolocation.getCurrentPosition()` (non-blocking).
        - Compute sunrise/sunset for today; **preferred:** lightweight lib **`suncalc`**; if not desired, fallback to a simple 7:00/19:00 schedule.
        - If permission denied or calc fails, use fallback schedule (env-overridable via `THEME_SUNRISE=07:00`, `THEME_SUNSET=19:00`).
        - Schedule the next switch using `setTimeout` and then re-schedule daily; also re-evaluate at app resume.
    - **UI controls:**
        - In the top-right overflow / settings, add **Theme** control (inline sheet or modal) with radio options: **System, Light, Dark, Auto (Sunrise→Sunset)**.
        - Add a small **Sun/Moon** quick-toggle button that cycles Light ↔ Dark (long-press opens theme sheet).
        - Show status hint under Auto: “Based on your local time; uses 7am/7pm if location isn’t shared.”
    - **Meta theme-color:**
        - Dynamically set `<meta name="theme-color" content="#...">` to match the header/background for better mobile status bar contrast (update on theme change).
    - **Persistence for verified users (nice-to-have):**
        - If user is verified, also store preference in DB (e.g., `humans.theme_mode` & optional `humans.theme_sunrise/sunset`), with endpoint `PATCH /api/me/theme { mode, sunrise?, sunset? }`.
        - On load, server can return `theme` in `/api/me` and client uses it instead of localStorage. (If not implemented now, localStorage is enough.)

🛠️ Implementation details (what to change)

A) Policy & config

Expose env-overridable policy (server config):

```
guestCharLimit    = env.GUEST_CHAR_LIMIT ?? 60
guestDaily        = env.GUEST_DAILY ?? 10
guestCooldownSec  = env.GUEST_COOLDOWN_SEC ?? 30
```

```
verifiedCharLimit = env.VERIFIED_CHAR_LIMIT ?? 240
verifiedPerMin    = env.VERIFIED_PER_MIN ?? 5
verifiedPerHour   = env.VERIFIED_PER_HOUR ?? 60
verifiedPerDay    = env.VERIFIED_PER_DAY ?? 200

WORLD_ID_APP_ID
WORLD_ID_ACTION
WORLD_ID_API_BASE   = env.WORLD_ID_API_BASE ??
'https://developer.worldcoin.org'
WORLD_ID_VERIF_LVL  = env.WORLD_ID_VERIF_LVL ?? 'orb'

DISABLE_WORLDID     = env.DISABLE_WORLDID ?? '0'
ENABLE_PERMIT2      = env.ENABLE_PERMIT2 ?? '0'

THEME_DEFAULT_MODE  = env.THEME_DEFAULT_MODE ?? 'system'  //
'light'|'dark'|'system'|'sun'
THEME_SUNRISE       = env.THEME_SUNRISE ?? '07:00'        // fallback
when no geo
THEME_SUNSET        = env.THEME_SUNSET ?? '19:00'
```

`GET /api/policy` must include the guest/verified limits (unchanged) and echo `themeDefaults: { mode, sunrise, sunset }`.

B) Guest post route — fix the guard

In `POST /api/messages`:

- **Branch by role early**.
  - **Guest path**: enforce **only** guest length/quota/cooldown; if all pass → create message.
  - **Verified path**: enforce verified limits; allow long text & verified features.
- Never emit “verification required” on the guest path.
- Log blocked reason codes (machine + human messages).

C) World ID Verify — end-to-end

- **Frontend:** use MiniKit **Verify** command for the incognito action; on success POST the proof to `/api/verify/worldid`, then update role in place and refresh policy.

- **Backend:** validate body; ensure `action` matches; hash `signal` if used; call **Cloud Verify v2** endpoint; on success write hashed nullifier (UNIQUE), set `wm_uid`, and return `{ ok:true, role:'verified' }`.

D) UX copy + landing

- Swap “Admin Access” → **Verify with World ID**.
- Landing card copy:
  - **Title:** “Guest access available”
  - **Body:** “Say hi with up to 60 characters (10/day). Verify with World ID to unlock full chat, stars, reports, and Work Mode.”
- Composer: live counter + mode pill text.
- Banners map to server codes (see Acceptance).

E) Content filter

- Relax repetition rule as described; keep emoji runs OK; still block obvious low-entropy spam.

F) Diagnostics

- `GET /api/worldid/diag` with `{ appId, action, apiBase, verificationLevel }`.
- `GET /api/debug/session` (non-secret) to show cookies seen + role for guest-cookie troubleshooting (webview quirks).

G) Theming files to add/update

- `client/src/theme/ThemeProvider.tsx` (context + effect to set `data-theme`, manage system listener, schedule sun timers).
- `client/src/hooks/use-theme.ts` (simple hook exporting `mode`, `setMode`, `activeTheme`).
- `client/src/index.css` (or `theme.css`) new variables for light & `[data-theme="dark"]` overrides.
- Header UI: add Theme quick-toggle (Sun/Moon icon) and Theme sheet (radio group with System/Light/Dark/Auto Sun).
- Update meta `<meta name="theme-color">` on theme change.

✅ Acceptance checks (please run & paste outputs)

Policy/Me

- `GET /api/policy` → `{ guestCharLimit:60, guestDaily:10, guestCooldownSec:30, ... , themeDefaults:{ mode:'system', sunrise:'07:00', sunset:'19:00' } }`.
- `GET /api/me` (fresh guest) → `{ role:"guest", theme?: { mode?: string } }`.

Guest flow

- As guest, post "hey everyone 👋" (≤60) → **200 OK**.
- Immediate second post → **429** `{ code:"guest_cooldown", wait≈30 }`.
- After 10 valid messages/day → **429** `{ code:"guest_daily_limit" }`.
- 61-char as guest → **403** `{ code:"guest_length_exceeded" }`.
- "Howdyyyyy" no longer blocked unless truly low-entropy spam.

Verify flow (Mini App)

- Tap "Verify with World ID" → World App verify drawer opens.
- On success: `/api/verify/worldid` → `{ ok:true, role:"verified" }`; UI flips to verified (240 chars; stars/report/Work Mode visible).
- Duplicate nullifier → **400** `{ code:"duplicate_nullifier" }`.
- `GET /api/worldid/diag` shows `{ appId, action, apiBase, verificationLevel }`.

Dark Mode

- Default mode follows **System** (if `THEME_DEFAULT_MODE=system`).
- Toggle quick Sun/Moon switches Light ↔ Dark; persists across reloads (localStorage).
- Theme sheet allows choosing **System / Light / Dark / Auto Sun**; selection applies immediately and persists.
- When **System** and device is Dark, UI is dark; switching device theme updates app live.
- When **Auto Sun**:
  - ○ If geolocation allowed: theme is **Light in daytime** and **Dark at night** by local sunrise/sunset.
  - ○ If geolocation denied: falls back to **07:00→19:00** schedule.
- `<meta name="theme-color">` updates to match background for both Light/Dark.

Observability

- Log summary line: `guest_accept=X guest_block_cooldown=Y guest_block_daily=Z verify_success=A verify_fail=B`.
- Log theme mode changes with: `theme_mode_changed { from:…, to:…, reason: click|system|sunrise|sunset }`.

Production sanity (unchanged)

- Single port in prod; Node serves build; no dev/HMR server in prod.

🔐 Secrets checklist (print status after build)

- `WORLD_ID_APP_ID` present?
- `WORLD_ID_ACTION` matches Dev Portal action?
- `WORLD_ID_API_BASE` default ok?
- If anything missing/mismatched, print a **“FIX SECRETS”** block with what to set.

🧭 Notes & design intent (for the agent)

- **Verify (incognito) vs OIDC:** we’re using Mini App **Verify** to gate human-only actions with no redirect. We may add OIDC/NextAuth later for full “account login”.
- **Wallet Auth later:** recommended in Mini Apps for stable identity (SIWE). Not required today; verify is sufficient to unlock features.
- **Friends/DM later:** we’ll reuse message infra with `dm_threads` & `dm_members`; not in scope today.
- **Custom emojis/Plus later:** keep behind flags (`EMOJI_SHOP_ENABLED=0`, `PLUS_ENABLED=0`) and reuse Room Rain points as currency; no UI today.

📦 Deliverables to paste back to the task log

1. Final **policy JSON** and **diag JSON**.
2. A short acceptance **test log** (guest OKs/blocks, verify success/fail, theme mode tests).
3. **Files changed** (paths only).
4. Any remaining TODO with a 1-line plan.