Aerune System Architecture Overview (English)

1. Basic Application Structure

- Platform: Electron (desktop app for Windows and macOS)
- Frontend: Vanilla JavaScript + HTML/CSS (no frameworks such as React or Vue)
- API client: @atproto/api (official Bluesky package)
- Architectural policy: Prioritizes lightweight performance and speed.
  Uses Event Delegation and DocumentFragment to minimize rendering overhead.
- Why not React/Vue?: Keep the app lightweight and reduce runtime overhead.
- Why Event Delegation?: Save memory by avoiding per-post/per-button listeners.

2. Module Structure (Role of Each JavaScript File)

Communication / Data Fetch Layer
- bsky-api.js (API wrapper)
  - Wraps BskyAgent from @atproto/api.
  - Handles login, fetching timelines, posting, likes/reposts, notifications, etc.
- view-loader.js (data shaping / view preparation)
  - Converts raw API responses into UI-friendly structures.
  - Builds threaded views (reply chains), prepares notification lists, loads profiles.

Rendering / UI Construction Layer
- post-renderer.js (post creation / rendering)
  - Builds a single post as DOM (or HTML) for display.
  - Handles image embeds, quote embeds, NSFW blur rules, and time rendering/caching.
- utils.js (utility toolkit)
  - Helpers such as linkification (linkify), rich-text rendering, and local image compression.

User Interaction / Action Layer
- actions.js (user actions)
  - Executes API operations after UI actions (Like, Repost, Bookmark, Follow, Block, etc.).
- navigation.js (navigation / history)
  - Manages navigation history (stack) and scroll position for Back/Forward behavior.

Controller
- renderer.js (main entry point)
  - Orchestrates modules, manages global state, switches views, sends posts.
  - Centralizes user input handling and event delegation.

Static Data
- constants.js (constants / translations / icons)
  - Stores i18n strings and SVG icon templates.
  - Keeps modules clean and prevents code bloat.

3. Key Design Points

3.1 Lightweight Design via Event Delegation
Attaching event listeners to every button (Like, Repost, etc.) scales poorly as posts increase.
Aerune uses a single listener at a higher-level container (or document) and triggers actions based on
data attributes (example: data-act="like"). This reduces memory usage significantly.

3.2 Optimistic UI Updates
UI updates immediately after user actions (Like/Repost/Delete) without waiting for the API response.
This keeps interactions responsive even under network latency. If the API fails, the UI can revert or show an error message.

3.3 Threaded Timeline Reconstruction
Replies are reorganized by tracing parent posts up to the API fetch limit, rebuilding:
parent -> child -> grandchild (and so on), then rendering in that order.
Additionally, the timeline loader can filter out unrelated non-followed conversations while keeping replies directed to the user.

Module Relationship (Overview)
[renderer.js (Controller)]
  |
  +-- bsky-api.js   (server communication)
  +-- view-loader.js   (data shaping)
  +-- post-renderer.js (DOM/HTML building)
  +-- actions.js   (user actions)
  +-- navigation.js   (navigation/history)
  +-- utils.js   (shared helpers)
  +-- constants.js   (i18n/icons/constants)