Aerune System Architecture Overview (English Version)

1. Application Structure

Platform: Electron (desktop app for Windows and Mac)
Frontend: Vanilla JavaScript + HTML/CSS (no frameworks such as React or Vue)
API Communication: @atproto/api (official Bluesky API package)
Architecture Policy: Prioritizes lightweight performance and speed. DOM is manipulated directly, using event delegation and DocumentFragment to minimize rendering overhead.

2. Roles of JavaScript Modules

[Communication / Data Layer]

bsky-api.js (API Wrapper)

- A wrapper around BskyAgent from @atproto/api.
- Handles login, post fetching, sending likes/reposts, retrieving notifications, etc.
- All communication with Bluesky's servers goes through this module.

view-loader.js (Data Processing & View Output)

- Converts raw API responses into structures suitable for rendering.
- Handles timeline threading (building hierarchical parent→child→grandchild structures).
- Also responsible for loading notifications and profile information.

[Rendering / UI Layer]

post-renderer.js (Post Builder)

- Constructs a single post as DOM elements.
- Handles embedding images, rendering quoted posts, NSFW blur logic, and relative time formatting with caching.

utils.js (Utility Functions)

- Contains helper tools such as linkify, rich-text rendering, and local image compression.
- A backstage module that supports other components.

[User Action / Interaction Layer]

actions.js (Post Actions)

- Executes API actions after a user interacts with buttons such as Like, Repost, Bookmark, Follow, Block, etc.

navigation.js (View Navigation & History)

- Manages navigation history and scroll position for the “back” operation.
- Functions as a lightweight router for the app.

[Controller]

renderer.js (Main Entry Point)

- The first script executed when the app launches.
- Coordinates all other modules.
- Manages global state (logged-in account, input text, selected images, etc.).
- Handles event delegation, screen switching, and sending posts.

[Static Data]

constants.js (Constants, Translations, Icons)

- Stores language data (i18n), SVG icon strings, and other constants.
- Prevents code bloat and keeps modules clean.

3. Core Architectural Features

(1) Event Delegation for Lightweight Performance
Instead of attaching addEventListener to every Like button or image, the app monitors clicks from a single location (document).
If the clicked element contains attributes like data-act=“like”, the corresponding action is triggered.
This drastically reduces memory usage.
Implemented in renderer.js under installDelegates.

(2) Optimistic UI Updates
When a user likes, reposts, or deletes a post, the UI updates immediately without waiting for the server response.
This creates a faster, more responsive experience than the web client.
Implemented in renderer.js inside the switch(act) block.

(3) Threaded Timeline Reconstruction
Replies in the timeline are reorganized by tracing parent posts upward to the API limit.
A structured list (parent → child → grandchild) is built before rendering.
Additionally, filters remove irrelevant conversations from non-followed users while keeping replies directed at the user.
Implemented in fetchTimeline within view-loader.js.