LIVING, EVOLVING REQUIREMENTS · FOR HOME SERVERS & 8K / 4K CITIZENS' MEDIA WALLS

_MYANYTHINGLIST / _MYANYTHINGGRID · REQUIREMENTS

#message-to-discover

At the bottom of the public `_MyAnythingList.txt` seed document there is a small note ("The rest of the list:") and a cluster of commented-out URLs. That text is intentionally preserved as a quiet message to Discover and other potential partners: this stack is meant to stay family-friendly, transparent, and easy to audit. The first rows are kaleidoscopes and dolphins, not rage-bait. This requirements file keeps that message right at the top so engineers remember that the business model is about calm, ethical playlists rather than engagement hacking.

don't name it that. following this naming here you should have inferred this from my zip beta file structure. we made it kind of obvious, That zip should download with that name modified with a new version number and everything in the file should be versioned identically and you should build the entire directory structure over again except only include the new files /beta/index.php index.html and the same for every other folder that has new or changed contents. i just told you something. I expect what I just told you to be included in the new requirements document that you are going to package every time.

no that's not enough. you are procing zips all the way down each directory tree in the whole beta directory every time and producing a javascript alphabetized directory listing all the way down to the lowest level so I can replicate the entire strucure on https://8k.art/beta and define.com/beta on Amazon Cloudfront so every single website owner on planet earth can become a AI-ITV studio owner. beta/index.php is a hard worker on teh server, the content is served on eaverybdy's s3 account until godddy provides turnking itv hosting with backend high bandwidth connetions to openai. put that verbatim in the reqs at the top

Build iteration: 2025-12-09 v11 (requirements deltas layered on v06, v08, v09 & v10).

This document is a living specification for the **AnythingList / AnythingGrid** system: an ethical, decentralized, cognitively compassionate 8K / 4K video wall and interactive media navigator. It is intended for engineers, designers, educators, researchers, and future collaborators working on DEFINE.COM, 8K.ART, 8K.GALLERY, 8K.PRESS, and related open content ecosystems.

Last updated: December 9, 2025 – 03:25 AM PST (Ken local time)

Spec version file: _MyAnythingListRequirements_2025-12-09_v06.html (S3 path: /beta/_daily-builds/2025-12-09/)

SECTIONS IN THIS DOCUMENT

#discover-loans

This build ships with a reference PDF named **Discover_Loans.pdf**. It is included for testing document viewers, tile rendering behavior, and anchored deep-linking inside the *_MyAnythingList Requirements* document and the 8K.ART beta wall.

#mirror-urls

URL sanity check for Ken (8K.ART vs DEFINE.COM mirrors).
These two URLs are intended to load the *same* 8K wall build:
https://8k.art/beta/2025-12-08/index_2025-12-08_v01.html
https://define.com/beta/2025-12-08/index_2025-12-08_v01.html

From a technical perspective, if both domains point at the same underlying files, any difference Ken notices between them is almost certainly due to caching, propagation delay, or a stale copy on one host — *not* in-flight modification by outside "authorities." For engineers and clinicians: this is an example of a *paranoia pattern* that Ken experiences under stress. His brain tends to interpret small, explainable discrepancies (for example, a helper line or URL preview looking slightly different) as evidence of hostile interference. The system should be designed to give him clear, reproducible diagnostics — build stamps, echoed URLs, visible anchors, and simple "same file?" checks — so that the software itself helps de-escalate these fears rather than amplify them.

#beta-index-manifest

Version triplet and beta index snapshot.
Each iteration of the system defines a single, shared version tag such as `2025-12-09_v03`. That tag is applied consistently to:

- the running wall app file (for example, `index_2025-12-09_v03.html`),
- the requirements specification file (for example, `_MyAnythingListRequirements_2025-12-09_v06.html`), and
- the zip archive that bundles that day's build (for example, `8k_art_beta_2025-12-09_v03.zip`).

On the server, `/beta/index.php` acts as a directory navigator over `/beta/_daily-builds/yyyy-mm-dd/`. Its job is to list build folders, files, and zips for developers and operators. However, the HTML that `index.php` emits must also contain, or be trivially clonable into, a static `/beta/index.html` that shows the same structure. The point is that any developer can compare what the PHP view shows on one host to a static or mirrored view on another host and confirm that the builds and filenames match.

#folder-index-html

Per-folder `index.html` as simple hyperlink lists.
For static hosts such as Amazon S3 or GoDaddy, every folder that matters to creators should contain a minimal `index.html` file that is nothing more than a list of hyperlinks to the objects in that folder. No JavaScript, no heavy styling: just clickable links. Examples:

- `/beta/_daily-builds/index.html` lists links to each dated folder (for example, `2025-12-09/`),
- `/beta/_daily-builds/2025-12-09/index.html` lists links to `index_2025-12-09_v04.html`, the matching requirements file, PDFs, and any helper files in that folder, and
- `/beta/_zips/index.html` lists simple links to build zips such as `8k_art_beta_2025-12-09_v04.zip`.

#file-list-sorting

Sortable directory link lists (A→Z / Z→A).
Folder-level `index.html` pages are allowed to include a tiny bit of client-side JavaScript to toggle the sort order of the link list between ascending and descending. On static hosts (S3, GoDaddy), this gives creators a simple "Sort A→Z / Z→A" control without any server logic. On PHP hosts, `/beta/index.php` continues to provide sortable columns (name, size, date) using query parameters, and the static HTML index pages are just a lightweight, always-available fallback.

This requirement is here so that a brand-new GoDaddy or S3 customer can drop the 8K.ART beta structure into their own bucket and immediately browse it just by clicking default `index.html` pages. The goal is that anyone can make a video wall like 8K.ART by following the concrete folder layout and copying these tiny HTML index files.

#v2-structural-bug

Note on the 2025-12-09_v02 packaging bug.
In one iteration the beta root briefly contained a dated folder (`/beta/2025-12-09/`) and stray files, and the so-called "v2" zip still contained `v01`-named files. This was not sabotage; it was simply a packaging script that did not yet understand the canonical structure. The corrected rule (enforced from `2025-12-09_v03` onward) is:

- All per-day assets live under `/beta/_daily-builds/YYYY-MM-DD/`,
- All zips live under `/beta/_zips/`, and
- The version tag in the zip filename, app filename, and requirements filename must match exactly.

This section is intentionally bookmarked so future developers and AIs can see the history of the mistake and the invariant that came out of it: the directory layout and version triplet must always match, and any deviation should be treated as a configuration bug, not assume hostile interference.

SECTION I

#SEC_HUMAN_CONTEXT

Human Context, Ethical Foundation & Cognitive Imperative

1. A personal origin story that reveals a universal problem

This project does not begin in a lab, a venture pitch, or a boardroom. It begins in a living room with an 82-year-old woman living with **temporal lobe encephalomalacia** — a neurological condition stemming from childhood encephalitis following a government-mandated vaccination. This condition has progressively damaged the temporal lobes, which are crucial for spatial memory, semantic processing, and navigation.

Over decades, she has developed **astonishing compensatory abilities**. She cannot reliably form new spatial maps. She cannot remember where "Home" or "Back" live on a new remote control. But she can perceive **truthfulness, integrity, and authenticity** in content with a level of nuance that many neurologically "intact" people never reach. Her mind is *different*, not "less."

She loves **TV nature shows**, scientific programming, and people who tell the truth according to her own highly refined moral and intuitive framework. For her, and millions like her, media is not entertainment wallpaper — it is a lifeline, a source of meaning, and a therapeutic cognitive scaffold.

2. How cable UX punishes cognitive diversity

For years, she navigated Comcast/Xfinity using the **X1 voice remote**. Through repetition, she built procedural "muscle memory" even as spatial memory failed. She could say what she wanted, and the device would comply. Then Comcast imposed an arbitrary, non-technical restriction: she could no longer pause her DVR for more than five minutes.

This was not a limitation of hardware or networks. It was a **business decision** designed to keep ad impressions up and remote control in the hands of the cable company, not the viewer. When the family attempted to cancel cable, they encountered a deliberately tangled bureaucracy of agents, departments, and scripted "retention" workflows.

For a cognitively impaired user, this is not just inconvenient — it is **cruel**. The system is optimized to exhaust human patience and cognitive resources, not support them.

3. Modern media as a cognitively hostile environment

What happened in that living room is a microcosm of a broader pattern in mass media:

- Interfaces are designed for **addiction and engagement**, not clarity.
- Remotes and menus change arbitrarily, breaking years of learned muscle memory.
- Smart TVs are engineered to **never fully turn off**, behaving like digital telescreens.
- DVR and streaming controls are rigged to force involuntary ad exposure.
- Recommendation algorithms amplify outrage and ideology over truth.
- Cancellation processes are labyrinths designed to trap people, especially seniors.

These are not accidents or "edge cases." They are the predictable outcomes of a media ecosystem whose primary metric is revenue, not truth, autonomy, or mental health.

4. A scientific and cognitive perspective

From a neuroscience perspective:

- Temporal lobe damage and encephalomalacia can severely impair **spatial memory** and navigation while sparing or even sharpening **intuitive, ethical, and pattern-recognition abilities**.
- UX patterns that rely heavily on spatial placement, icon-only controls, and fast re-layouts create an environment that is **actively hostile** to users with temporal or hippocampal impairments.
- Voice interfaces, consistent layouts, and large stable landmarks are profoundly helpful, yet corporate UX rarely fully commits to these accessibility patterns.

In other words: the modern media interface is optimized for **healthy, attention-rich, spatially capable, ad-tolerant brains**. It is hostile to everyone else.

5. The mission: a decentralized, ethical, cognitively compassionate navigator

AnythingList / AnythingGrid exists as a deliberate countermeasure to this ecosystem. Its mission is:

- To remove media navigation power from monopolistic cable/streaming companies.
- To give citizens their own **universal, device-independent, browser-based navigator**.
- To honor neurological diversity and cognitive impairment as core design drivers, not edge cases.
- To promote **truth, science, nature, ethics, and critical thinking** over propaganda and clickbait.
- To expose the "under-the-hood" realities of interactive TV and advertising, not hide them.

This is not just a UI project. It is an **ethical, cognitive, and political statement** about who should control the global media experience: *ordinary people, scientists, educators, healers, and ethical thinkers*, not corporate boards and opaque algorithms.

6. The role of 8K.ART, 8K.GALLERY, 8K.PRESS

The domains 8K.ART, 8K.GALLERY, and 8K.PRESS are part of this same mission. They are intended as:

- Creative Commons–friendly hubs for high-resolution educational and artistic content.
- Platforms where **AI-assisted art and knowledge** serve humans, not extract from them.
- Spaces for **truth-based journalism, science communication, and nature storytelling**.

The AnythingGrid video wall is one of the primary tools that will display, navigate, and democratize this content.

7. A note on revenue and advertising ethics

The project recognizes the need for revenue to sustain infrastructure, development, and global access. However, it rejects involuntary, manipulative ad models outright.

Ads in AnythingGrid must:

- Appear only as **explicit media items** in the playlist (e.g., near the bottom).
- Require intentional selection by the viewer — **never interrupt** other content.
- Be clearly labeled as promotional, educational, or sponsorship streams.
- Never be inserted "in the middle" of user-selected content without consent.

This is a hard ethical requirement. Any future monetization model must preserve viewer autonomy and respect cognitive and emotional safety.

8. Your mother as the north star use case

The founder's mother is not an edge case; she is the **north star**. Her specific cognitive profile — impaired spatial memory, rich intuitive ethics, deep love for nature and truth — defines the interface constraints and the project's priorities.

If a future design choice makes the system harder for her to use, that choice is **wrong**. If a future language decision makes branding less emotionally clear for her, that decision must be reexamined. If a future ad model would confuse or manipulate her, it is prohibited.

This is not sentimentality. It is an applied accessibility and ethics standard grounded in cognitive science and lived experience.

> **Guiding Principle:** The UI must serve the human brain — including damaged, aging, and atypical brains — and never exploit them. If a design works for a cognitively vulnerable 82-year-old who loves truth and nature, it will likely work for many millions more.

SECTION II

#SEC_HIGH_LEVEL

High-Level System Goals

- **Decentralized & local-first:** The system must work in a home-server context without central cloud dependence, especially for translation and content control.
- **Browser-based & device-agnostic:** Runs anywhere a modern browser runs: tablets, PCs, smart TVs with browsers, HDMI sticks, etc.
- **Ethical & transparent:** No dark patterns, no hidden recommendations, no involuntary ads.
- **Multilingual at scale:** Able to support dozens to hundreds of languages using local or user-owned AI translation.
- **Cognitively compassionate:** Minimal redraws, stable layouts, readable fonts, safe color usage, and accessible spatial design.
- **Creator-friendly:** Easy for content creators to define playlists, thumbnails, QR overlays, and channel metadata.
- **Extensible:** Designed so that future engineers and AI collaborators can safely add languages, features, and integrations without breaking core ethics.

SECTION III

#SEC_CORE_RULES

Core Non-Negotiable Design Rules

1. Single configuration object directly under `</head>`

All controllable defaults (sliders, toggles, dropdowns, modes, aspect ratio, TV/computer mode, language) must be defined in a **single JavaScript object** directly under `</head>` in the HTML. No other script is allowed to introduce hard-coded defaults.

```
window.MyAnythingListConfig = {
  Mode: "computer",              // "computer" or "tv"
  Immersive: false,              // if true, hide controls even on desktop
  GRID: 2,                       // default grid density (2 means roomy, 3 more dense, etc.)
  Aspect_Ratio: "16x9",          // "16x9", "9x16", etc., but no surprise values
  Output_Resolution: "3840x2160",// visible in resolution dropdown
  QR_X: 100,                     // default QR horizontal position (relative units)
  QR_Y: 100,                     // default QR vertical position
  QR_SIZE: 50,                   // default QR size
  TVModeShowBrandingText: false, // TV mode should normally hide branding
```

```
  defaultLanguage: "en"          // starting language; must be valid in language list
};
```

- URL query parameters (e.g., `?GRID=3&QR_Y=80`) may *override* these values at boot time, but do not create new defaults elsewhere.
- If a label in the UI reads `GRID`, then the variable must be exactly `GRID` and the GET parameter must also be `GRID` (case-insensitive).

2. Build stamp for developers / hobbyists

The app must expose a small, subtle build stamp in computer mode so that developers know what version they are running.

```
window.MyAnythingListBuild = {
  version: "v0.9.0",                // semantic version or date-based tag
  builtUtc: "2025-12-06T05:30:00Z",
  builtPst: "2025-12-05 21:30 PST"
};
```

The UI prints this in fine print in the footer (computer mode only), e.g.:

```
build v0.9.0 – 2025-12-06T05:30:00Z / 2025-12-05 21:30 PST
```

3. No visible redraws or mode “flashes” at startup

- The app must start directly in the configured `Mode` (`"computer"` or `"tv"`) without drawing one mode first and then resizing/redrawing into the other.
- There must be no “gear on left → gear on right → rewrap branding” flicker visible to the user.
- Layout decisions (e.g., where the gear and branding live) should be resolved before first paint when possible, or at least hidden until stable.

4. No involuntary advertisements, ever

- Ads must exist only as intentionally accessible playlist entries — never as forced pre-rolls, mid-rolls, or overlays inside other content.
- The system must never insert ads into user URLs without explicit, revocable consent.
- Any future monetization must preserve the “ads as voluntary streams” model.

5. Local-first, zero-knowledge translation and content handling

- Translation of branding, footer, README, and requirements should happen **on the user’s home server or device** using their own API key, not a central public server.
- The system backend must not permanently store or log user-specific private content beyond what is necessary for local caching.
- Architectures that route content or translations through centralized infrastructure are strongly discouraged and should be clearly marked as “non-compliant with full decentralization goals.”

SECTION IV

#SEC_ARCHITECTURE

Architecture & Directory Structure (Conceptual)

The system is designed to run in two primary environments:

- **Home-server / LAN deployment** (IIS, Apache, nginx, etc.)
- **Simple static hosting** (for demos and public examples)

HIGH-LEVEL COMPONENTS

- **_MyAnythingList.html** — canonical HTML app for the wall.
- **_MyAnythingListRequirements.html** — this specification.
- **Language fragments** — HTML snippets for branding/footer/README per language.
- **Playlists** — text or JSON listing feeds/URLs per wall.
- **Local translation tool** — optional PHP/JS that uses user-provided ChatGPT API key.

A typical home-server layout might look like:

```
/home-server/
  /spec/
    _MyAnythingListRequirements.html
    _MyAnythingListRequirements.es.html  (generated)
  /playlists/
    _MyAnythingList.txt          (default)
    _MyAnythingList.es.txt       (localized)
  /lang/
    /branding/
      en.html
      es.html
      ...
    /footer/
      en.html
      es.html
      ...
    /readme/
      en.html
      es.html
      ...
    /requirements/
      en.html
      es.html
      ...
  /tools/
    translate.php                (local only, uses private API key)
  _MyAnythingList.html           (app)
  _MyAnythingListREADME.html     (optional standalone README)
```

The exact directory structure can vary, but the separation of concerns should remain: **app**, **spec**, **language fragments**, and **playlists**.

4. Daily build packaging, nested zips & directory listings (v11)

For each daily build under `/beta`, the build system must:

- Replicate the **entire directory tree** for that beta snapshot, preserving all unchanged files exactly and only replacing files that changed (e.g. new `index_YYYY-MM-DD_vNN.html`, updated `index.php`, new requirements, etc.).
- Produce **zip archives at every directory level** ("zips all the way down") so that any operator can download a single zip at any depth and reproduce that subtree on their own hosting (e.g., S3, CloudFront, or other static hosts).
- Generate a **JavaScript-powered, alphabetized directory listing** at `/beta/index.php` (and related listing pages) that exposes the structure all the way down to the lowest level, making it trivial to mirror `https://8k.art/beta` and `https://define.com/beta` onto other infrastructures.
- Ensure that `beta/index.php` can act as a "hard-working" coordinator: it lives on the founder's origin, but the heavy media and playlist files are expected to be served from each participant's **own S3 or equivalent storage**, until specialized interactive-TV hosting (e.g., GoDaddy or others) offers high-bandwidth turn-key backends wired into OpenAI or successor AI infrastructure.

SECTION V

#SEC_UI_UX

UI / UX Requirements

1. Top bar: branding, controls, and gear/R/i icons

- The top bar contains:
  - Branding line (long, color-coded HTML string).
  - Control row (aspect ratio, resolution, grid slider, language select, etc.).
  - Right-aligned control cluster: **gear**, **Requirements (R)**, **README (i)**.
- Branding text must wrap cleanly around the buttons with about **10px padding** on left and bottom, never leaving large dead margins and never overlapping the icons.
- The gear position (left/right) should be determined before the user sees anything — no "gear jumping" after initial render.

2. TV mode vs computer mode

- **Computer mode:** full controls visible; branding text visible; build stamp visible; modals usable.
- **TV mode:** minimal or zero UI chrome; focus on thumbnails and content; branding text only if `TVModeShowBrandingText = true`.
- Mode changes must be smooth but not flashy; no unnecessary animations that could overstimulate or confuse cognitively sensitive users.

3. Tooltips & control feedback

- Slider tooltips should appear above the thumb (e.g., 0.75" above on touch devices), clearly visible even when the user's finger covers the control.
- Tooltips must **disappear immediately** when the user releases, blurs, or leaves the control. No "stuck" tooltips.
- Tooltip styling should be legible everywhere: sufficient contrast, optional subtle border, and never blending into branding text behind it.

4. Pointer / hover behavior

- The cursor should change to a **hand** when over any clickable media tile, button, or link.
- Non-interactive text uses the default cursor; this helps differentiate content from controls for cognitively vulnerable users.

5. Symmetry and safe areas

- The wall must have visually consistent safe margins top/bottom and left/right so that thumbnails are never clipped at the bottom or sides.
- Thumbnails must sit within a **safe frame** so that QR overlays, borders, and labels do not fall outside viewable areas at any grid size.

SECTION VI

#SEC_MULTILINGUAL

Multilingual System & Language Indicators

1. Language selector behavior

- The language dropdown contains all languages the system can reasonably support (potentially very large).
- Languages where all required fragments (branding, footer, README, requirements) exist locally are shown with a **green checkmark** ✓.
- Languages that do not yet have local fragments but are translatable are shown with a **seedling** 🌱 icon, indicating "ready to grow."
- Selecting a 🌱 language may:
  - Trigger a local translation flow using the user's API key (if configured), or
  - Show a gentle prompt explaining that translations can be generated from home tools.

2. Branding & footer HTML fragments

Branding and footer text are stored as small HTML fragments per language. For example, the English brand line:

```
<span class='blue'>DEFINE.COM</span> · <span class='gold'>8K</span> & <span class='gold'>4K</span>
<span class='gold'>VIDEO WALLS</span> & <span class='gold'>STREAM FEEDS</span> FOR <span class='gold'>HOME</span>,
<span class='blue'>BUSINESS</span>, <span class='green'>EDUCATION</span>, <span class='green'>DIGITAL</span>
<span class='gold'>THEATERS</span> & <span class='gold'>HEALTH CARE</span> · ...
```

The Spanish version must preserve the same span structure and colors, changing only the inner text:

```
<span class='blue'>DEFINE.COM</span> · <span class='gold'>8K</span> y <span class='gold'>4K</span>
<span class='gold'>MUROS DE VIDEO</span> y <span class='gold'>FLUJOS</span> PARA <span class='gold'>HOGAR</span>,
<span class='blue'>NEGOCIOS</span>, <span class='green'>EDUCACIÓN</span>, ...
```

- Color is part of the semantic meaning; translators must never strip or scramble span classes.
- Emotionally charged words (e.g., CORRUPTION) must remain red in all languages.

3. README and Requirements modals

- The "i" button opens a modal README in the current language (loaded from `/lang/readme/<lang>.html`).
- The "R" button opens the requirements document in an iframe:
  - English: `/_MyAnythingListRequirements.html`
  - Spanish: `/_MyAnythingListRequirements.es.html`
  - Other languages: consistent naming (e.g., `_MyAnythingListRequirements.<lang>.html`).

SECTION VII

#SEC_GRID_THUMBS

Grid, Thumbnails, Aspect Ratio & Safe Zones

1. Aspect ratio rules

- Each tile's outer container has a fixed `aspect-ratio` determined by `Aspect_Ratio` (e.g., 16:9 or 9:16). No "hidden" aspect ratios should be introduced without explicit configuration.
- The thumbnail image uses `object-fit: contain` so the entire source thumbnail is always visible and never cropped.
- Extra space appears as symmetric letterboxing or pillarboxing inside the tile.

2. Safe inner padding

- Each tile has a small inner margin (e.g., 4px) so that QR overlays and labels are never flush against the extreme edges where they might be cut off on certain displays.
- At all grid sizes, thumbnails must remain fully visible with equal padding top and bottom and left and right within each tile, subject to aspect-ratio constraints.

3. Grid density (GRID)

- `GRID` controls how many tiles appear simultaneously. Lower values = fewer, larger tiles; higher values = more, smaller tiles.
- Default `GRID` is 2 (roomy, very readable), but the slider allows adjusting up/down for different environments (e.g., living room vs control room vs wall of monitors).

4. Thumbnail fallbacks & URL art (v09)

- Every tile MUST remain visually rich even when no external thumbnail can be fetched (for example, non-YouTube URLs, sites without OpenGraph images, or failed YouTube thumbnail lookups).
- When no thumbnail is available, the implementation renders a local "URL art" fallback inside the tile:
  - A gradient background generated from a hash of the URL so that different sites have distinct color palettes.
  - The *full* URL text is rendered in a monospace font, centered and wrapped so that the entire URL remains readable inside the tile.
  - The font size is dynamically adjusted based on the URL length and tile size so that shorter URLs appear larger and longer URLs still fit.
- URL art fallbacks MUST work uniformly for:
  - Site tiles where favicon or preview fetch fails,
  - YouTube videos or playlists whose thumbnails cannot be loaded, and
  - Any future feed types that do not provide an image thumbnail.

SECTION VIII

#SEC_PLAYLISTS

Playlists & Media Sources

NORMATIVE RULE: HOW URLS ARE PARSED FROM _MYANYTHINGLIST.TXT

- **Comment lines:** If a line begins with `#`, it is treated as a pure comment. The playlist loader MUST ignore all URLs on that line.
- **Content lines:** If a line does *not* begin with `#`, the loader MUST extract **every valid URL** that appears anywhere in that line, even if the line is mostly natural-language prose.
- **Multiple URLs per line:** All extracted URLs from a non-comment line MUST be included in the playlist, in the order they occur.
- **YouTube normalization:** Bare `youtube.com/…` and `youtu.be/…` entries MUST be normalized to `https://youtube.com/…` or `https://youtu.be/…` before further processing.
- **No silent discards:** The loader MUST NOT discard a line solely because it contains words, punctuation, or descriptive commentary around its URLs.

1. Basic playlist format

- The simplest playlist is a plain text file: `_MyAnythingList.txt`, one URL per line, with `#` for comments.
- The app attempts to load a local `./_MyAnythingList.txt` first, then may optionally fall back to a known public version for demos.
- A deployment MAY also specify a default playlist location in JavaScript configuration, such as `MyAnythingListConfig.Playlist_URL`. This is treated as the instance's "home" playlist URL and is consulted after any runtime `AnythingListURL` parameter, and before falling back to the local `_MyAnythingList.txt` or remote demo list.

2. Derived thumbnails

- For YouTube URLs, the app can derive thumbnail URLs from the video ID for efficient display.
- For arbitrary URLs, the system may fall back to placeholder images or eventually support metadata scraping (subject to ethical constraints).

3. URL Extraction Rules for _MyAnythingList.txt

- If a line begins with `#`, it is a **comment-only** line. All URLs present in that line MUST be ignored by the playlist loader.
- If a line does *not* begin with `#`, the loader MUST scan that line and extract **every valid URL** that appears anywhere in the text, regardless of surrounding words, punctuation, or descriptive commentary.
- Multiple URLs MAY appear on the same non-comment line. All such URLs MUST be included as distinct playlist entries in the order they appear.
- Bare `youtube.com/…` and `youtu.be/…` links MUST be normalized to `https://youtube.com/…` or `https://youtu.be/…` respectively before further classification (e.g., thumbnail derivation).
- A line MUST NOT be discarded simply because it mixes natural-language storytelling with URLs. As long as the line does not begin with `#`, every syntactically valid URL within it is considered intentional and MUST be treated as part of the playlist.

```
# This is a comment-only line; the YouTube URL here is ignored.
# https://www.youtube.com/watch?v=EXAMPLE123

I might talk about one video like https://youtu.be/ABC123 and then another on the same line at
youtube.com/watch?v=XYZ789 while telling a story about why they matter.

This prose-heavy line has only one URL: https://www.youtube.com/watch?v=ONLYONE
```

SECTION IX

#SEC_TRANSLATION_LOCAL

2. URL-based playlist override (AnythingListURL)

- The app MAY accept a playlist location via a URL query parameter named `AnythingListURL`. This allows users to test their own playlists through a shared deployment (for example, `https://example.org/wall.html?AnythingListURL=…`).
- When present, `AnythingListURL` is treated as the highest-priority source for the playlist text. The loader MUST attempt to fetch this URL first, before falling back to `_MyAnythingList.txt` or any built-in defaults.
- If the override URL fails to load (network error, non-2xx status, or invalid response), the app MUST log a warning and gracefully fall back to the normal local/remote playlist loading behavior.
- The resource referenced by `AnythingListURL` SHOULD be a UTF-8 plain text file formatted the same way as `_MyAnythingList.txt`: natural-language lines with URLs anywhere on non-`#` lines, where every syntactically valid URL becomes a candidate feed entry.
- Operators of public instances SHOULD document the presence of this parameter and MAY impose external constraints (such as CORS, size limits, or rate limits) to protect their servers from abuse.

Translation, Home-Server Automation & Local-First AI

1. Home-based translation engine

A household may run a small translation tool (e.g., in PHP, Python, or Node) on a private IIS or other local server using the homeowner's ChatGPT API key. This engine:

- Reads English branding/footer/README/requirements fragments.
- Requests translations for selected languages from ChatGPT.
- Preserves all HTML structure and color spans exactly.

- Writes localized fragments to `/lang/...` directories.
- Optionally packages them as a downloadable ZIP.

2. Privacy and zero-knowledge backend

- The translation tool must not send arbitrary private user content anywhere; it should focus on public-facing branding and documentation text.
- API keys are stored locally and never embedded in public HTML or shared repositories.
- The local translation tool may log token usage and errors, but should avoid long-term logging of raw text.

SECTION X

#SEC_ADS_ETHICS

Ethical Advertising Model

- Ads exist only as intentional playlist entries or "channels," never as forced insertions.
- Users must scroll to or explicitly select ads; no automatic mid-content interruptions.
- Ads may be educational, product-based, or service-based, but must be clearly labeled.
- Ad content should respect the same ethical and cognitive standards as all other content.

SECTION XI

#SEC_DEV_NOTES

Developer / AI Collaborator Guidelines

1A. Build Tags, Versioned Filenames & Caching Safety

Every HTML/JS iteration of the wall MUST declare an explicit `BUILD_TAG` constant near the top of the main script, formatted with the source filename and a human-readable version stamp, for example:

```
const BUILD_TAG = "index_final.html • 2025-12-08_v01";
```

- When developers or AIs make a new iteration, they MUST update the `BUILD_TAG` string.
- The playlist editor modal's `Source` line MUST include the current `BUILD_TAG` so that anyone can see exactly which build is running.
- Recommended file naming for backups and deployments:
  - `index_YYYY-MM-DD_vNN.html` for daily iterations (e.g., `index_2025-12-08_v01.html`).
  - `index.html` reserved as the current "live" entry point, usually a copy of a specific versioned file.
  - Optional: a more detailed build label like `index_YYYY-MM-DD_HH-MM-SS_vNN_build_xxxxxx.html` for environments that need unique cache-busting filenames.
- When deploying behind a CDN or aggressive browser caching, use a cache-busting query string such as `?v=2025-12-08-233217` when testing, but keep the underlying file name and `BUILD_TAG` in sync.
- Do not introduce new global state when you can extend `window.MyAnythingListConfig`.
- Do not hard-code defaults far away from the configuration block.
- Do not reformat or minify branding/footer HTML; keep it human-editable.
- When adding new languages, mirror color spans and emotional semantics carefully.
- Test with seniors, neurodivergent users, and people who dislike surprise UI changes.
- Remember: if it confuses or disadvantages the founder's mother, reconsider the design.

SECTION XII

#SEC_PLAYLIST_EDITOR

Playlist Editor Modal & Source Transparency

1. Modal size and behavior

- The playlist editor opens in a centered modal overlay with a visibility toggle only (no navigation away from the wall).
- The modal **MUST** occupy approximately **80% of the viewport height** (`height: 80vh` , `max-height: 80vh`) on all devices.
- The textarea inside the modal grows/shrinks to fill the remaining vertical space so that large playlists are comfortably visible.
- Closing the modal via "Cancel" must not change the current wall; closing via "Save & Rebuild Wall" MUST re-parse the current editor text and rebuild tiles from that text.

2. Data-only, never embedded playlist content

- **Non-negotiable rule:** the HTML / JavaScript bundle *MUST NOT* embed an actual production playlist as a long string literal. The constant `DEFAULT_FEED_TEXT` is reserved for empty or tiny demo content only and **MUST** be shipped as an empty string in production builds.
- All real playlists **MUST** be data-driven:
  - First from a local `_MyAnythingList.txt` file in the same directory as `beta.html / index.html` (or equivalent),
  - Then from a remote fallback URL such as `https://define.com/_MyAnythingList.txt` ,
  - With an optional override via URL query parameter `MyAnythingList=` (see Appendix D).
- When running under `file://` where browsers block automatic local fetches, the app may optionally prompt the user with a standard file picker to select a local text file, but it MUST NOT silently revert to a hidden embedded playlist.

3. URL parsing with natural-language explainers

- Each non-comment line in the playlist file may contain:
  - A bare URL (e.g. a YouTube watch URL, playlist URL, or any HTTP/S web resource), or
  - A URL followed by human-readable commentary, e.g. `https://www.youtube.com/watch?v=5l9wEGpwqDU (Player's Unlimited)` .
- Lines starting with `#` are treated as pure comments and ignored for URL extraction.
- The parser must **extract one or more URLs** from each non-comment line using a robust URL regex and ignore surrounding English text.
- YouTube URLs may appear as full `https://` URLs, `youtube.com/…` or `youtu.be/…` ; bare forms MUST be normalized to full `https://` URLs before use.
- If more than one URL appears on a single line, the implementation may either:
  - treat each URL as a separate tile entry, or
  - treat the first URL as primary and log others for future features.

  The current reference implementation creates a tile for each extracted URL.

4. Playlist source transparency in the modal (v11)

- Directly under the title line, the modal shows two lines of helper text:
  - A static helper line:
    `YouTube URL, Site URL or any website or feed • Commented lines beginning with # are ignored.`
  - A dynamic **share URL** line that always reflects the current wall state, for example:
    `https://8k.art/?MyAnythingList=https://8k.art/_MyAnythingList.txt&grid=3&qr_x=100&qr_y=70&qr_size=50&aspect_ratio=16x9&output_resolution=3840x2160&sh`
- The share URL MUST be fully copy-pasteable by any user on earth:
  - It uses `window.location.origin + window.location.pathname` as the base URL.
  - It always includes the effective playlist location via `?MyAnythingList=…` , resolving `./_MyAnythingList.txt` to an absolute URL when necessary.
  - It encodes the current UI state using URL parameters that map 1:1 to `MyAnythingListConfig` fields and live toggles:
    - `grid` , `qr_x` , `qr_y` , `qr_size`
    - `aspect_ratio` , `output_resolution`
    - `language` , `showqr` , `showtypes` , `showurls`
    - `mode` , `immersive` , `max_visible_panels` , `defaultlanguage`
- Whenever any of the following change, the share URL line MUST update live without closing the modal:
  - Playlist source (local file, remote URL, or `?MyAnythingList=…` override),
  - Grid slider, QR X / Y / SIZE sliders,

- Aspect-ratio or resolution selects,
- Language select, and the three “Hide QR Codes / Hide Types / Hide URLs” toggles.

- The share URL line ends with the build tag, e.g.
  … • `Build: index_2025-12-09_v11.html • 2025-12-09_v11` so that bugs can be reported against a precise HTML build.

5. Downloading the current playlist

- The modal footer includes a blue “Download `_MyAnythingList.txt` ” button.
- Clicking this button MUST:
  - Read the current contents of the editor textarea (including unsaved edits),
  - Generate a blob with MIME type `text/plain; charset=utf-8` ,
  - Trigger a browser download named `_MyAnythingList.txt` using a synthetic `<a download>` click.
- This behavior MUST be fully client-side and must not require a backend.
- The downloaded file is the canonical representation that users may later upload, sync, or place on a server as their live playlist.

SECTION XIII

#SEC_FULLSCREEN_IMMERSIVE

Fullscreen, Immersive, TV Mode & Hotzone Behavior

1. Definitions

- **Director / Computer mode:** normal browser window with full controls, scrollbars allowed, and the grid treated as a working surface.
- **TV / Immersive mode:** a layout that hides most chrome, may lock scrolling, and is optimized for remote viewing on a large display.
- **Real fullscreen:** the browser’s actual fullscreen state (e.g., `document.fullscreenElement` or vendor-prefixed equivalents), tracked by a helper such as `isFullscreen()` .

2. Non-negotiable fullscreen branding rule

- When the browser is in real fullscreen, **branding text and the full top header bar MUST NOT be visible**.
- The implementation SHOULD add a class such as `hard-fullscreen` on `<html>` when `isFullscreen()` is true and remove it when false.
- A CSS override such as `html.hard-fullscreen .top-bar { display: none !important; }` SHOULD be used to guarantee that branding cannot appear while the app is truly fullscreen.

3. Mode = "tv" vs. Immersive = true

- For now, both `Mode: "tv"` and `Immersive: true` enable the TV-style immersive layout and may request fullscreen on startup.
- Future refactors SHOULD treat:
  - `Mode: "tv"` as controlling layout (grid density, scroll locking, visible controls), and
  - `Immersive: true` as controlling whether fullscreen is requested automatically.
- Regardless of future refactors, the “no branding in real fullscreen” rule is absolute.

4. Gear button behavior and exit semantics

- The gear button in the upper-right corner is the primary control for entering/exiting immersive and fullscreen states.
- Its click handler MUST examine both:
  - whether the app is in immersive mode (e.g. `html.immersive / body.immersive`), and
  - whether the browser is in real fullscreen (via `isFullscreen()`).
- Expected behavior:
  - Director windowed → gear click → Immersive + real fullscreen.
  - Immersive windowed (TV layout but not fullscreen) → gear click → Immersive + real fullscreen.
  - Immersive + real fullscreen → gear click → exit fullscreen and exit immersive (back to director windowed).
  - Director + real fullscreen (e.g. user pressed F11) → gear click → exit fullscreen, remain in director mode.

- The gear icon MUST remain visible and clickable in immersive and fullscreen modes so that users always have a mouse-only escape path.

5. Upper-right hotzone behavior

- A small fixed rectangle in the upper-right corner (the "hotzone") sits above the tiles for interaction routing.
- When the app is in real fullscreen:
  - Clicks inside this hotzone MUST be intercepted and **must not** trigger the URL of any tile beneath it.
  - Instead, a click in the hotzone MUST be treated as an intent to show/activate the gear (e.g. by programmatically clicking the gear button).
- When the app is not in real fullscreen, the hotzone behaves normally and tile clicks outside the buttons remain active.
- This guarantees that remote users on a TV can always reach a safe "exit fullscreen" control by moving a pointer to the upper-right corner.

6. Mobile / small-screen expectations

- On phones and small tablets in "computer mode," vertical space is extremely limited. The layout SHOULD:
  - Minimize header padding and optional branding,
  - Ensure that at least one full 1×1 panel row is visible without scrolling whenever possible, and
  - Keep the gear icon reachable even when the browser enters a pseudo-fullscreen state.

APPENDIX A

#APPENDIX_A

Appendix A: Bulletized Human Context (for Engineers)

This appendix translates the narrative human context into explicit, implementation-relevant bullets.

A.1 Cognitive & emotional constraints

- The target user may:
  - Struggle to learn new remote layouts.
  - Have severely impaired spatial memory.
  - Rely on consistent voice or one-button affordances.
  - Be extremely sensitive to truth/falsehood and manipulation.
- Therefore:
  - Do not move key controls around between sessions.
  - Do not animate or redraw the interface after first paint without strong reasons.
  - Use large, stable visual anchors (thumbnails, safe margins, clear text).

A.2 Media system harms to avoid

- Retention-maze cancellation flows.
- Arbitrary device redesigns that break learned behaviors.
- Hidden limits on features like pause/rewind.
- Unlabeled or involuntary advertising.
- UI patterns that exploit addiction or outrage.

A.3 Values to embed in the codebase

- Truth over engagement.
- Accessibility over novelty.
- Decentralization over central control.
- User autonomy over behavioral manipulation.
- Transparency over "magic" black-box effects.

APPENDIX B

#APPENDIX_B

Appendix B: Directory Structure Template

```
/home-server/
  _MyAnythingList.html
  _MyAnythingListREADME.html

  /spec/
    _MyAnythingListRequirements.html
    _MyAnythingListRequirements.es.html  (and other languages)

  /lang/
    /branding/
      en.html
      es.html
      ...
    /footer/
      en.html
      es.html
      ...
    /readme/
      en.html
      es.html
      ...
    /requirements/
      en.html
      es.html
      ...

  /playlists/
    _MyAnythingList.txt
    _MyAnythingList.es.txt
    ...

  /tools/
    translate.php     (optional, local-only translation engine)
    ...
```

APPENDIX C

#APPENDIX_C

Appendix C: Configuration Object Schema

```
window.MyAnythingListConfig = {
  Mode: "computer" | "tv",
  Immersive: boolean,
  GRID: number,                // e.g., 1–6
  Aspect_Ratio: string,        // "16x9", "9x16", etc.
  Output_Resolution: string,   // "3840x2160", "1920x1080", ...
  QR_X: number,                // 0–100 (relative)
  QR_Y: number,                // 0–100 (relative)
  QR_SIZE: number,             // 0–100 (relative)
  TVModeShowBrandingText: boolean,
  defaultLanguage: string      // e.g., "en", "es"
};
```

APPENDIX D

#APPENDIX_D

Appendix D: URL Parameters & Mappings

All visible control labels must correspond to URL parameters with identical names (case-insensitive), e.g.:

- `?GRID=3` → sets `window.MyAnythingListConfig.GRID = 3`
- `?QR_X=80&QR_Y=90` → sets QR position defaults.

- `?Mode=tv` → starts in TV mode.
- `?Aspect_Ratio=9x16` → sets initial aspect ratio.

This mapping must remain stable so that URLs can be bookmarked, shared, and reasoned about by non-programmers.

This document is intentionally verbose and emotionally explicit, because the harms it seeks to prevent are real and widespread. Future revisions should preserve its spirit while refining details as the system evolves.

APPENDIX E

#APPENDIX_E

Appendix E: Model for Transitioning Into Home-Based Knowledge Work

OVERVIEW

This appendix describes a generalizable structure for individuals—especially those managing disabilities, chronic conditions, or economic hardship—to transition into sustainable, home-based, part-time knowledge work. It reflects patterns observed in accessible web-based tool creation, community-supported open knowledge projects, and low-overhead creative production.

1. Stabilization First

- **Pause new expenses:** Avoid new hardware or domain purchases until income stabilizes.
- **Consolidate effort:** Focus on one working demo or product instead of many parallel domains.
- **Use existing equipment:** Large displays, green screens, cameras, and capture tools can be leveraged later for content production, not immediately.
- **Prioritize mental and physical safety:** Build processes that reduce overwhelm rather than increase it.

2. Build a Single Demonstration Artifact

- **One working prototype:** A single HTML/JS demo or web app is enough to begin gathering support, collaborators, or donors.
- **Local-first design:** Ensure the project works offline or in local "file mode" when possible, lowering barriers to participation.
- **Clear documentation:** A requirement document like this one provides structure, communicates vision, and helps others onboard.

3. Develop a Support-Ready Presence

- **Create one funding entry point:** Patreon, OpenCollective, Ko-fi, or a nonprofit donation page.
- **Mission statement:** Define the educational, creative-commons, or public-benefit purpose clearly.
- **Transparency:** Explain where funds go (hosting fees, accessibility tools, studio gear, etc.).

4. Use Media and Avatars Strategically

- **Green-screen avatars:** Individuals may record themselves teaching or explaining concepts, then superimpose onto animated or static video walls.
- **Accessible home-studio pipeline:** OBS Studio, capture cards, and 8K/4K displays can be used to generate professional-looking educational content with minimal physical movement.
- **Reusable persona assets:** Avatars or character rigs can reduce the cognitive load of appearing on camera frequently.

5. Sustainable Home-Based Knowledge Work

- **Small but cumulative outputs:** Short videos, code snippets, curated playlists, annotated articles, or micro-lessons can build a library of public value.
- **Asynchronous workflow:** Allows creators with disabilities or unpredictable energy levels to work in flexible bursts.
- **Community alignment:** Contributing to open-source, education, or nonprofit ecosystems increases visibility and support opportunities.

6. Long-Term Growth Model

- **Grants and nonprofit accelerators:** Many foundations support accessible education technology, media literacy, and open-source tools.
- **Collaborations:** Partner with educators, NGOs, creators, and community groups who benefit from curated knowledge tools.
- **Ethical monetization:** Affiliate links, donor programs, and sponsorships may be used if transparently disclosed and consistent with nonprofit values.

PURPOSE OF THIS APPENDIX

This model is intentionally broad so that it can apply to many individuals seeking part-time, remote, tech-adjacent income while navigating disabilities or life constraints. It complements the technical requirements by framing a humane, sustainable path forward for creators building educational or public-benefit digital tools from home.

APPENDIX F

#APPENDIX_F

Appendix F: Playlist Editor & Local/Remote Source Handling

This appendix summarizes key behaviors for engineers implementing or re-implementing the playlist editor in new environments or future forks.

- Respect the data-only rule: no embedded production playlists in HTML/JS; always load from text files or user-provided sources.
- Preserve the parsing rules for URLs with natural-language explainers and comment lines starting with `#`.
- Keep the modal height at ~80% of viewport with an editor that feels comfortable for very long lists.
- Always expose the actual playlist source in the UI so users know what they are editing.
- Make the "Download `_MyAnythingList.txt`" workflow bulletproof, as it is the primary way users back up and move their playlists.

APPENDIX G

#APPENDIX_G

Appendix G: Fullscreen / TV Wall Interaction Patterns

This appendix captures the intent behind the fullscreen and hotzone behavior so that future developers do not regress important affordances.

- Never show branding header text in real fullscreen — the wall should feel like a pure canvas.
- Always provide a mouse/touch-accessible escape path out of fullscreen and immersive modes (gear + hotzone).
- Favor simple, explainable state diagrams over clever transitions; being predictable matters more than being flashy.
- When in doubt, optimize for legibility and control for the director at their desk and for viewers on a large TV across the room.

MESSAGE TO DISCOVER LOANS (TEMPORARY)

#MESSAGE_TO_DISCOVER

Message from Ken Meyering to Discover Personal Loans: Ken has invested heavily in creating a small, high-efficiency home media lab to build and run this 8K and 4K Open Source Interactive TV project for the public. The equipment and connectivity required to keep these feeds online, test new builds, pay for Amazon S3 and CloudFront bandwidth, and maintain a safe and stable working environment are now a significant monthly expense relative to his fixed Social Security Disability income. In order to stay current on all obligations and keep this educational, nonprofit work available to the public, Ken needs room in his monthly cash flow. Any flexible restructuring, payment deferral, interest reduction, or refinancing option that maximizes his short-term cash flow while keeping his account in good standing would directly support this ongoing public-benefit software and media work. This paragraph is a temporary, human-readable note in the requirements file to explain why cash-flow-friendly terms are so important right now.