ChatGPT

Wildcat.chat Platform Homepage Design Plan

Objectives and Branding

The goal is to create a **unified landing page** at **wildcat.chat** that introduces the Wildcat platform and its Twitch tools (ChatSage, ChatVibes, and the AI Theme Chat Overlay) in a cohesive, enticing way. This homepage should **tie together all sub-services** (currently split across `app.wildcat.chat`, `tts.wildcat.chat`, `docs.wildcat.chat`, and a GitHub Pages overlay) under a consistent Wildcat brand. We will ensure the design is flexible for future renaming of the bots to match Wildcat branding. Key objectives include:

- **Consistent Wildcat Identity:** Prominently feature the Wildcat name/logo (e.g. the kaomoji mascot used on the docs site) and a tagline like *"AI-Powered Twitch Bot Platform."* All bot names will be presented as part of Wildcat (e.g. "Wildcat Chatbot (ChatSage)" or simply "Wildcat Chatbot") so that if names change later, the branding remains consistent.
- **Enticing First Impression:** Use modern, slick design elements that the user favors – **an animated static background** and **3D "floating" buttons with sharp corners and drop-shadows** – to immediately grab attention [1] [2]. The aesthetic should feel techy and streamer-friendly, aligning with Twitch's vibe.
- **Clear Navigation:** Provide easy links to each tool (ChatSage, ChatVibes, Overlay) and documentation from one place. A visitor should instantly see what Wildcat offers and where to click to learn more or try the tools.

By unifying the look and feel across the homepage and the sub-apps, we reinforce the Wildcat brand and make the platform feel like one product rather than disparate parts.

Design Aesthetic and Theme

The homepage will adopt the same design language already used in the ChatSage and ChatVibes UIs for a seamless experience. Key design choices include:

- **Animated Static Background:** Implement a full-browser **canvas background** that simulates TV noise or subtle moving static, just like the bot UIs have done [1]. This adds a tech-inspired texture without distracting from content. The canvas will be fixed (`position: fixed; top:0; left:0; width:100%; height:100%`) and behind all content (z-index -1) so it stays in the background [3]. We'll reuse or adapt the existing static background script from ChatVibes/ChatSage for consistency.

- **Floating 3D Buttons:** Use the distinctive button style from ChatVibes – **sharp rectangular buttons with a 2px solid border and offset drop shadow** [2]. These will serve as the main calls-to-action (links to apps, docs, etc.). On hover, they will **shift upward and cast a larger shadow** to emphasize an interactive "pressable" 3D effect [4]. This gives a playful, tactile feel to the links. We'll apply this style to all major links (e.g. "Launch App" or "View Docs" buttons), ensuring the CSS (background-

color, border, box-shadow, transition) matches the framework used in tts.wildcat.chat for easy maintenance.

- **Color Scheme and Fonts:** Stick to a **clean light theme** as default: white or light-gray card backgrounds with black text and borders (as seen in the current UIs). Accent colors can come from Twitch's palette (e.g. the Twitch purple `#6441A5` for primary buttons or highlights) to evoke the streaming context. For example, the "Login with Twitch" or key action buttons can have purple background with white text, similar to how ChatVibes styles its Twitch login button with purple [5]. Use high contrast for readability: black text on white cards, and ensure dark mode friendliness (the CSS can include `prefers-color-scheme` adjustments as in the docs site). Typography will use the **Atkinson Hyperlegible** font family (already used in Wildcat docs/UIs) for clarity, with perhaps *Cabin Condensed* or similar for accent or headings [6] [7]. These fonts give a modern, approachable look and align with the existing branding.

- **Card and Container Style:** Content sections will be presented on **"cards" or panels with subtle borders and drop shadows**, echoing the ChatVibes dashboard container style. For instance, we can use a class like the existing `.container` which provides a white background, rounded corners, a 2px solid border, and an 8px offset shadow for a floating effect [8]. This way, feature sections feel like floating panels above the static background, maintaining that slick 3D aesthetic throughout the page.

- **Unified Iconography/Imagery:** Introduce minimalistic icons or illustrations for each tool (e.g. a chat bubble icon for ChatSage, a speaker icon for ChatVibes, and a paint palette or screen icon for the Overlay). These can be simple SVGs in the Wildcat color style. If available, incorporate the Wildcat kaomoji face as a subtle branding element (e.g. in the header or footer). Visual consistency is key: icons should share a style (outline vs filled) and match the sharp, geometric feel of the site. Any imagery should not clash with the static noise background – likely we'll keep graphics simple and possibly semi-flat with a slight shadow to fit the theme.

- **Overall Layout:** Use plenty of whitespace (or rather *"empty space"* given the background isn't solid) around elements so the page doesn't feel cramped. The design should guide the eye from the hero section down through the features. We'll center most content and likely use a **single-page scroll** layout for now (since it's a simple landing page), which also works well on mobile. The style will be **responsive** by nature (we can use CSS grid/flexbox and the existing CSS variables for spacing and breakpoints). On smaller screens, cards will stack vertically and buttons will remain large and tap-friendly.

Homepage Structure and Content

We will structure the homepage into clear sections for easy scanning. Below is the proposed layout from top to bottom:

1. Hero Header

Appearance: The hero is a full-width banner section, center-aligned, with the animated static background visible behind. The Wildcat logo/mascot and title text will sit here, likely in a container with a translucent or white backdrop for contrast.

- **Title:** A large, bold heading introducing Wildcat. For example: **"Wildcat: AI Tools for Twitch Streamers."** This immediately tells visitors what the platform is about. The font can be the mascot or primary font in a heavier weight. If using the kaomoji logo, the text can be alongside it [9] [10].
- **Tagline:** A one-liner subtitle that highlights the value proposition. E.g., *"Knowledgeable chatbots, realistic TTS, and smart overlays – all in one platform."* This tagline should be concise and engaging, underscoring what makes Wildcat special (AI-powered, enhances stream interaction). In the docs site, a tagline like "AI-Powered Twitch Bot Platform" is used [11], which we can adapt here.
- **Call-to-Action Buttons:** Prominently below the title/tagline, include 1-3 large buttons for immediate actions. For example:
- **"Try ChatSage"** – linking to the ChatSage app (app.wildcat.chat) for logged-in usage.
- **"Try ChatVibes TTS"** – linking to tts.wildcat.chat.
- Optionally **"Live Demo"** or **"Watch Overlay in Action"** – if a preview of the overlay is available. Each button will use the 3D style (white background or an accent color with black border) described earlier [2]. They should be large enough to stand out as the next step. If there are many tools, we could consolidate to a single "Explore Wildcat Tools" button that scrolls the user to the features section below, but having separate buttons for each bot gives quick access.
- Possibly include a **Twitch OAuth prompt** if appropriate (for example, a "Login with Twitch to Get Started" button). However, since access is invite-only at the moment, we might instead have a "Request Access" link in smaller text. The main buttons can still lead to the apps (which then show the invite-only message if not approved).

The hero section sets the stage with branding and a clear directive on where to go next. All of this is contained in a visually striking banner with the static noise behind to grab attention.

2. Platform Overview

Right below the hero, a brief introduction in text form will explain what Wildcat is in a friendly tone. This can be a short paragraph (2-3 sentences) along the lines of:

> "**Wildcat** is a suite of AI-driven Twitch bot tools designed to elevate your stream's interactivity. From answering chat questions intelligently to reading messages aloud with natural TTS and generating on-brand chat overlays, Wildcat brings advanced AI to your channel with minimal setup."

This overview gives context and can mention that the platform currently includes multiple components. We should emphasize that all tools are **seamlessly integrated** and maintained under the Wildcat platform (so users know it's one ecosystem). If relevant, note that it's currently in beta/invite-only, to set expectations. The tone should be enthusiastic and focused on benefits to streamers (engagement, fun, ease of use).

Visually, this section might just be text on the page (centered or slightly left-aligned) possibly atop a subtle card background for readability. We can apply the container style here if needed for contrast (e.g., a semi-

transparent dark overlay behind text if the static background makes it hard to read, or simply use a white container with 90% width).

3. Features/Tools Section

This is the meat of the homepage – showcasing **each major Wildcat tool** with a bit of info and a link. It could be laid out as a **3-column grid** (for desktop) or vertical stack (on mobile). Each tool gets a **card** or panel as follows:

- **ChatSage (Wildcat Chatbot):**
- *Description:* One or two sentences explaining ChatSage. For example: "**ChatSage** is an AI-powered chatbot that lives in your Twitch chat. It can answer viewer questions, fetch information from the web, play trivia and mini-games, and even translate or role-play – all automatically in chat." We'll highlight unique features gleaned from docs (contextual responses, web search, games, etc. [12]).
- *Visual:* Perhaps an icon of a speech bubble or brain to represent knowledge. If we have a small screenshot of it in action (though ChatSage is text-based in chat, so maybe not visually distinctive), we might skip screenshots and stick to icons/illustrations.

- *CTA:* A button like **"Go to ChatSage Dashboard"** or **"Try ChatSage"**. If the ChatSage web UI requires login, the button would bring them to the Twitch OAuth login page of ChatSage. Alternatively, a **"Learn More"** link to documentation (the docs site section for ChatSage commands) could be here, but since we want to entice use, the primary action might be to launch the app. We can include both: a styled button for "Launch" and a smaller text link for "Docs".

- **ChatVibes (TTS Bot):**

- *Description:* "**ChatVibes** gives your chat a voice. This text-to-speech bot reads chat messages and alerts aloud in real time using natural-sounding AI voices. Engage your audience with hundreds of voice options, even generate AI music on command! Integrates with channel points and Bits for monetization." This encapsulates its main selling points (TTS with 300+ voices, music generation, etc. [13]). Emphasize that it's controlled via a web dashboard for easy setup.
- *Visual:* An icon of a speaker or waveform graphic could represent TTS. Possibly a small waveform animation or speaker icon next to the title.

- *CTA:* **"Open TTS Dashboard"** (leading to tts.wildcat.chat) as the big button. Also perhaps a **"View ChatVibes Docs"** link beneath it for more info on commands and setup.

- **AI Theme Chat Overlay:**

- *Description:* "**Wildcat Chat Overlay** is a customizable Twitch chat overlay with **AI-generated themes**. Describe any aesthetic (e.g. *'cyberpunk city'* or *'cozy forest'*) and let the AI design a matching background image and style for your chat. Includes an easy visual editor for colors, fonts, and more, plus multiple display modes (standard chat window or pop-up messages). No login required – just plug into OBS." This highlights the unique AI theme generator feature [14] [15] and ease of use (the quick web UI that provides a URL for OBS).
- *Visual:* Possibly a small preview image of an example themed chat (if available from the screenshots). For instance, we could show a thumbnail of the overlay interface or a before/after of a chat with an AI-generated background [16] . If not, an icon of a paint palette or monitor can be used.

- *CTA:* **"Try the Overlay Designer"** button linking to the GitHub Pages URL (or in the future, perhaps `overlay.wildcat.chat`). Because this tool doesn't require Twitch auth, we can directly send users there. Also include a "Learn More" link to the overlay's README or docs if needed.

Each of these feature cards will use a consistent style: a **headline** (tool name, maybe with small "Wildcat [Name]" branding), a short descriptive text, and one or two action links. We'll enclose each in a container/card with the 3D shadow style so they pop off the page [8]. The cards can have equal height if possible for a neat grid (CSS grid or flex with matching height). There should be some margin or padding between cards (the existing CSS spacing variables like `var(--spacing-medium)` can be used to keep it consistent). On hover, perhaps the whole card can slightly raise or just rely on button hovers inside.

To make this section **slick**, we could add a bit of animation: e.g., as the user scrolls to this section, each card could fade-in or slide-up slightly (using a simple CSS animation or intersection observer). This subtle motion will make the page feel more dynamic.

4. Documentation and Support

After the tools, have a section for **Resources** or **Next Steps**:

- **Documentation:** A short blurb like "Want to dive deeper? Check out our docs for full command lists, setup guides, and advanced tips." Followed by a **"Wildcat Docs"** button or link. Since docs are on a separate subdomain, ensure the link is obvious. We might reuse the style of the nav bar or an accent button for "Documentation". (In fact, the top navigation might already have a Docs link, but a callout here ensures new users see it.)

- **Community/Contact:** If there's a way to contact or request access (since currently invite-only), mention that here. For example: "**Request Access:** Wildcat is currently in beta. Interested streamers can get in touch here to request an invite." This can be plain text or a small button "Contact Us". We should use the same button style for consistency if making it a button.

- Possibly **GitHub Links:** If the user is open to it, we can link to the GitHub repos for these projects for transparency (especially since some are public). But this might be optional on the main homepage not to overwhelm non-technical users. Perhaps just a discreet footer note "Open Source on GitHub" with icons.

This section doesn't need to be large – maybe two or three columns or a simple centered text. It's mostly to ensure users have somewhere to go if they need help or more info.

5. Unified Navigation & Footer

To make the homepage feel integrated with the docs and apps, we should implement a **common navigation bar** at the top of the page (and possibly use it on docs as well). The docs site already has a top nav with links to Home, ChatSage Docs, ChatVibes Docs, Contact [17]. We can mirror that on the homepage: a slim navbar with the Wildcat logo on left and links on the right: *Home, ChatSage, ChatVibes, Overlay, Docs, Contact*. Each of those (except Home) would point to the appropriate sub-site or anchor. The nav bar should use the same styling as in docs – a fixed top bar with a bottom border and the menu items styled as small 3D buttons [18] [19] (which they are in the unified stylesheet). This not only looks polished but also reinforces

that all these sections are part of one site. On mobile, we'd include the hamburger toggle that the docs have (which reveals a vertical menu).

Finally, a **footer** can sit at the very bottom with minimal info. It could simply say "© 2025 Wildcat.chat" and maybe repeat a few key links or social contacts. Given much is covered in nav, the footer can be minimalistic. If there's a contact email or Discord, it can go here. The footer text can be small and perhaps on a solid background (or continue the static background if readability is okay).

Implementation Steps and Considerations

To achieve the above design in a maintainable way, here's the plan for implementation:

1. **Unify the CSS Framework:** First, refactor and extract the common styles from ChatVibes and ChatSage into a single **Wildcat CSS** (which can be similar to the existing `unified.css` used in docs [20]). This stylesheet will contain the core variables (colors, spacing, fonts) and base styles (reset, typography, link/button styles, container styling) that all Wildcat web UIs share. By doing this, both the homepage and the app UIs can include this file and instantly get the same look. We saw that ChatVibes has a maintainable CSS setup using CSS variables and a modern reset [21] [22] – we will apply that here.
2. **ChatSage Migration:** Update the ChatSage web UI (app.wildcat.chat) to use the new unified styles. For example, ensure it uses the same `.button` classes or `<a>` styling for its links and the same container classes as ChatVibes. Remove any outdated or duplicate CSS from ChatSage that conflicts. Essentially, ChatSage's `main.css` should be brought in line with ChatVibes' `styles.css` conventions (like using the common 3D button styles [23] [24] instead of separate rules). This migration not only makes ChatSage look consistent but also eases future changes – we'll have one place to tweak the design.

3. The **homepage** will also include this unified CSS. Since the docs already have `unified.css`, we might expand it or add a separate `home.css` for any homepage-specific tweaks. The goal is that things like fonts, colors, and button styles are pulled from the same source for consistency.

4. **Build the HTML Structure:** Create the homepage as a static HTML (perhaps served via Firebase Hosting or GitHub Pages on wildcat.chat). Structure it according to the sections outlined:

5. A `<header>` for the navigation bar (with the Wildcat logo and menu links). We can reuse the markup and classes from the docs site nav [25] [26]. This includes the mobile toggle button with the same classes and JavaScript (the docs likely have a `sidenav.js` or similar for the toggle functionality). By copying this, we ensure identical behavior. Update link URLs appropriately (Home can scroll to top or just `#`, ChatSage link might point to the app or an anchor section on the homepage if we decide to just scroll, but since the app is separate, a direct link is fine).
6. A `<canvas id="staticCanvas">` as the first element in the body (the canvas for background). Include or inline the static background script (likely the same `static-background.js` used in ChatVibes – we saw a reference to it in index.html [27]). This script generates the noise effect on the canvas. We'll ensure to initialize it on page load. Also include any required polyfills or small tweaks for performance (the existing script likely covers that).
7. The **hero section** markup: could be a `<section class="hero">` containing an `<h1>` for title, a `<p>` for tagline, and a div for the buttons. We might wrap the text in a div with class like `"hero-`

`content"` and style it to have some padding/margin-top to sit below the nav. The buttons can use the anchor with class `"button"` (and perhaps additional classes like `"primary"` for styling, or even reuse `.twitch-login` class style for a purple button if we want the Twitch-colored CTA).

8. The **overview paragraph** can be a simple `<p>` or small `<section>` after the hero, using standard text classes.
9. The **features** can be a `<section id="features">` with a container that uses a CSS grid. For each feature, use a `<div class="feature-card">` (we'll apply the container styles here, or a specific class that extends container). Inside, an `<h3>` for the tool name, a `<p>` for description, a list for key points if needed (like we could bullet a few standout features as done in docs [28] [13]), and the action links (which can be `<a class="button">Launch</a>` and maybe `<a class="docs-link">Docs</a>` styled appropriately). We'll make sure these cards are keyboard accessible and the links have `target="_blank"` if going to external pages (like documentation PDF or such, though here all are same domain except contact form).
10. The **docs/contact section** can be another `<section id="resources">` with maybe an `<h2>` like "Additional Resources" and paragraphs with links. We can style those links as inline or small buttons as needed.
11. The **footer** is a `<footer>` tag with a small container. Possibly just text or a horizontal rule then text. We don't need heavy styles here, maybe just center the text and give it a lighter color.

Throughout this HTML, we'll use the classes defined in our unified CSS (like `.container`, `.button`, `.site-navbar`, etc.) to get the desired styling out of the box, rather than writing new CSS. This ensures maintenance is low and style stays consistent if the CSS framework updates.

1. **Apply Styling & Responsive Design:** Once the HTML is in place, apply the CSS:
2. Include the `reset.css` (if not already in unified) to normalize styles across browsers [29].
3. Include the unified stylesheet. This gives us base font family, colors, etc., automatically. We'll verify that the body has the correct max-width and centering (the apps set a max-width of 1400px and center the content [30], which is good to keep content from stretching too wide on large screens).
4. Add any homepage-specific CSS. For example, we might want `.hero` to have extra padding-top (to center vertically) or a larger font-size for the title than default `<h1>`. We can either inline a small `<style>` for these or extend unified.css for homepage. We should also style the feature cards grid: maybe using something like `.features-grid { display: grid; grid-template-columns: repeat(auto-fit, minmax(300px, 1fr)); gap: 2rem; }` – in fact, the docs index.html already uses a similar snippet for the documentation cards [31]. We can borrow that.
5. Responsive adjustments: The grid approach with `auto-fit/minmax` will naturally stack the cards on narrow screens, but we'll test on mobile widths. The nav bar from docs likely already has a mobile mode (burger menu toggles display of `.site-navbar-menu`). We need to ensure to copy the corresponding CSS/JS for that (for example, the docs might hide the menu and show the toggle at smaller widths via CSS, which should be in `components.css` or similar). Bringing over those styles ensures the nav works on mobile. For the hero text, use relative units or CSS clamp to scale the font for smaller screens if needed (or just let it wrap). Buttons can stack vertically on mobile if side by side is too cramped – we can use media queries or simply place them in block format.
6. Test the **hover and active states** of buttons and links to ensure the 3D effect works (the unified CSS covers `:hover`, `:active` transformations [32] [33]). Also test the color contrast of text over the background – if the static noise is too busy, we might slightly darken/lighten the background behind text by using a semi-transparent overlay. One approach: give the body or section a subtle background color with some opacity (e.g., a translucent white) to soften the noise behind content.

Alternatively, increase the density of the static so it reads more like a textured background than individual moving dots, for a cleaner look. We want to keep the effect but not at the expense of readability.

7. **Performance:** The static background canvas might be resource-intensive; ensure the implementation is optimized (like updating at a modest FPS, or possibly using a CSS animated GIF/noise pattern as fallback). The homepage can afford it since it's not super heavy otherwise, but it's worth considering for slower devices. If needed, we can detect `prefers-reduced-motion` and disable the animated background, falling back to a static image.

8. **Integrate with Hosting and Links:** Coordinate with the domain setup:

9. Set up wildcat.chat domain to serve this new homepage (likely via Firebase Hosting if the user is using Firebase for others, as indicated by presence of `.firebaserc` in some repos). Possibly the docs and apps are also on Firebase; we can add a rewrite so that the root domain shows the homepage, and docs remain on docs.wildcat.chat.
10. Update any references in the sub-apps to point back to the main site if needed (for instance, the docs "Home" link currently goes to docs home, but we might want it to go to wildcat.chat main page once it exists). This ensures seamless navigation – a user on docs can click "Home" and get to the marketing homepage.

11. Since the overlay is on GitHub Pages (detekoi.github.io), we'll link out to that for now. In the future, consider moving it under wildcat.chat (maybe as overlay.wildcat.chat via CNAME or embedding it). But for this plan, a link is sufficient. We should just note it opens in a new tab to not fully navigate away.

12. **Finalize Content and Messaging:** Work with the team (or user) to refine the text on the homepage:

13. Make sure the descriptions of ChatSage, ChatVibes, and the overlay are accurate and enticing. Possibly incorporate any catchy phrases or branding (for example, if "Wildcat" has a meaning we can play on, do so).
14. Ensure that if the bots are renamed down the line, the content is written such that it's not a huge overhaul. We might use generic terms like "Wildcat Knowledge Bot" in the headline and mention "(formerly ChatSage)" in smaller font for now. The plan would be to quietly drop the old names when ready. Using the Wildcat name prominently now sets the stage for that.
15. Double-check that any claims (like "300+ voices" or features) are up-to-date with the latest state of the project. Given the rapid development, keep statements somewhat general but attractive. For example, instead of a specific number that could change, say "hundreds of voices".

16. If available, include one-line **testimonials or quotes** from beta users (this could be a nice touch if the user has any feedback quotes to display). These could scroll or just be static blurb in quotes to build trust. Not necessary, but worth considering if time permits.

17. **Testing and Feedback:** Once implemented, test the homepage in multiple scenarios:

18. **Desktop and Mobile browsers:** Ensure the layout adapts nicely. The button hover effect should degrade gracefully on touch devices (no hover on mobile, but active state might show on tap). The static background should not cause scrolling or overflow issues (we keep it fixed).

19. **Navigation flow:** Try coming from the homepage into the ChatSage or ChatVibes app and back, to see if the style transition is smooth. After migrating ChatSage's CSS, it should look similar to the homepage (same fonts, etc.), which will help. If there's a stark difference, adjust accordingly.
20. **Performance:** Confirm the page loads quickly. Since it's mostly static content plus one canvas animation, it should be fine. Optimize images if any are used (like compress any background logo SVG or icons).

21. **Accessibility:** Check that colors have sufficient contrast (the black text on white is good; the purple on white for buttons is readable, etc.). Ensure all interactive elements are keyboard accessible (the links and buttons should be naturally, and the nav toggle button too). Add `aria-labels` where appropriate (the nav toggle already had one in docs: "Toggle navigation" [34]). Also, consider adding `prefers-reduced-motion` media query to minimize animation for those who opt out (e.g., maybe don't animate the background or use a static image if that setting is on).

22. **Launch and Iterate:** Deploy the homepage and gather any user feedback. See if visitors understand the platform from the info given. If needed, we can tweak content or add elements (for example, a short comparison table if people want to know the difference between the bots at a glance, or a video demo link). The design framework is flexible for expansion. As new features or products are added to Wildcat, we can easily add another card to the features section due to the modular grid design.

By following these steps, we leverage the existing **maintainable CSS framework** and design elements from ChatVibes (and the unified docs style) to build a homepage that is not only visually appealing but also consistent and easy to keep up-to-date. The result will be a **slick, modern landing page** with a static noise backdrop and floating interactive buttons – a style that reflects Wildcat's tech-savvy personality and grabs visitors' attention from the start. Users arriving at wildcat.chat will immediately recognize the branding and have clear paths to explore the ChatSage bot, the ChatVibes TTS, the overlay, or documentation, all while enjoying a cohesive look and feel across the entire Wildcat platform. [2] [20]

---

[1] README.md
https://github.com/detekoi/chatsage-web-ui/blob/522da39a20e44b07015b7bb71ff2408f94972916/README.md

[2] [3] [4] [5] [6] [8] [21] [30] [32] [33] styles.css
https://github.com/detekoi/chatvibes-web-ui/blob/b3a72d0fb0e88f4b3c999aa0204616acad7313ce/public/css/styles.css

[7] [18] [19] [20] [22] unified.css
https://github.com/detekoi/wildcat-docs/blob/cdee519ba5fdf1ef908e9fd07f9d80779f69c30d/public/styles/unified.css

[9] [10] [11] [12] [13] [17] [25] [26] [28] [31] [34] index.html
https://github.com/detekoi/wildcat-docs/blob/cdee519ba5fdf1ef908e9fd07f9d80779f69c30d/public/index.html

[14] [15] [16] README.md
https://github.com/detekoi/compact-chat-overlay/blob/36f8f5ba964ec95c31ef5e41a09b4613ce4010da/README.md

[23] [24] main.css
https://github.com/detekoi/chatsage-web-ui/blob/522da39a20e44b07015b7bb71ff2408f94972916/public/css/main.css

[27] [29] index.html
https://github.com/detekoi/chatvibes-web-ui/blob/b3a72d0fb0e88f4b3c999aa0204616acad7313ce/public/index.html