Integrating NodeBB Comments into a Quartz Static Website

1. Introduction

Quartz is a modern static site generator designed for speed and extensibility, particularly appealing to users within the Obsidian ecosystem.[1] Its primary function is to transform Markdown-formatted content into fully functional websites, emphasizing a streamlined development experience.[2] NodeBB, on the other hand, is a contemporary, open-source forum software built on Node.js, renowned for its real-time discussion capabilities and a robust plugin architecture that allows for extensive customization.[5] The user's objective is to integrate the dynamic commenting features of NodeBB into a static website generated by Quartz, thereby enhancing their static content with interactive discussions. This report will explore the architectural nuances of both platforms, investigate potential integration methods, discuss the challenges and considerations involved in such a setup, present alternative commenting solutions, and conclude with a recommended approach to achieve the user's goal.

2. Understanding Quartz

- Architecture and Build Process:
  Quartz, an open-source static site generator crafted with NodeJS, is specifically tailored for individuals utilizing Obsidian for note-taking and knowledge management.1 Its fundamental operation involves converting Markdown files, the cornerstone of Obsidian's content, into a complete website.1 The process is initiated when a user executes the npx quartz build command within their project directory.10 This command triggers a sequence of events managed by NodeJS, starting with the execution of the bootstrap-cli.mjs file.10 This initial script is responsible for employing esbuild, a fast JavaScript and TypeScript transpiler and bundler, to convert the Quartz codebase, written in TypeScript, into standard JavaScript.10 Notably, the esbuild configuration is enhanced by the esbuild-sass-plugin, enabling the processing of SCSS files for website styling.10
  During the development phase, Quartz establishes two distinct servers to facilitate the creation and preview of the website.[10] A WebSocket server, operating on port 3001, manages hot-reloading, ensuring that any changes to the content or configuration are instantly reflected in the development environment.[10] Simultaneously, an HTTP file server, typically running on port 8080, serves the actual static website files, allowing developers to preview their site locally.[10] The parsing of Markdown files is a central part of Quartz's build process. It involves a series of transformations facilitated by a plugin-based system.[10] This includes reading the file, applying text transformations defined by plugins, generating a URL-friendly slug from the file path, parsing the Markdown content using remark-parse, applying further Markdown-specific transformations, converting the Markdown to HTML using remark-rehype, and finally,

applying HTML-specific transformations.[10] This multi-stage process ensures that the content is accurately and consistently processed according to the defined configurations and plugins.

The final stage of the build process involves emitting the website files using emitter plugins.[10] These plugins gather any necessary static resources and, for HTML files, perform an additional transformation step.[10] They convert the parsed HTML, which is represented as an Abstract Syntax Tree (AST) using Hast, into JSX (JavaScript XML) syntax using hast-util-to-jsx-runtime, specifically configured for the Preact runtime.[10] This JSX is then rendered into static HTML strings using preact-render-to-string, a process that focuses on the structural aspects of the page rather than interactive Preact components.[10] Additionally, Quartz handles CSS and JavaScript assets efficiently.[10] CSS files are minified and transformed using Lightning CSS, which also adds vendor prefixes for browser compatibility and performs syntax lowering.[10] JavaScript code is bundled and strategically split into two categories: beforeDOMLoaded scripts, which are placed in the <head> of the HTML to load critical functionality early, and afterDOMLoaded scripts, inserted at the end of the <body> to handle non-critical or deferred actions.[10]

Insight: The fact that Quartz is built upon NodeJS, a JavaScript runtime environment, creates a foundational compatibility with NodeBB, which is also built using NodeJS. This shared technology stack could potentially simplify the integration process. Furthermore, Quartz's well-defined build process, particularly the structured parsing and transformation of content, suggests that custom plugins or modifications could be introduced to facilitate the embedding of dynamic content from external sources like NodeBB.

- Plugin System:
  Quartz's architecture is highly modular, with a robust plugin system that enables developers to extend and customize nearly every aspect of the site generation process.1 This system allows for the modification of content at various stages of the build, from parsing Markdown to emitting the final HTML files.10 Plugins in Quartz are primarily categorized into three types, each serving a distinct purpose in the content pipeline 1:
    - **Transformers:** These plugins operate on the content itself, allowing for transformations such as reformatting text from one markup language to another (for example, Markdown to HTML) or manipulating the structure and syntax of the content.[1]
    - **Filters:** Filters provide a mechanism for selectively including or excluding content files based on predefined criteria. These criteria can often be based on metadata associated with the content, such as properties set within the YAML frontmatter of Markdown files.[1]
    - **Emitters:** Emitter plugins are responsible for taking the processed content and generating new output files. This can involve creating aggregated files, transforming content into different formats, or any other operation that results in the creation of new files within the generated website.[1]

The specific details and implementation of Quartz's plugin system are intentionally described in greater depth within a dedicated guide aimed at developers who wish to create their own custom plugins for the platform.[10] This detailed documentation allows for a high degree of control and customization over the site generation process.**Insight:** The existence of this flexible plugin system within Quartz is a significant advantage for the user's goal of integrating NodeBB comments. A custom Quartz plugin could potentially be developed to directly interact with the NodeBB API, fetch comments related to specific pages, and embed them into the generated static HTML. Alternatively, plugins could be used to modify the build process to include JavaScript code snippets that handle the embedding of NodeBB's comment widget.

- Content Delivery and Client-Side Scripting:
  As a static site generator, Quartz's primary strength lies in its ability to deliver website content with exceptional speed and efficiency by serving pre-rendered HTML files directly to the user's browser.1 This approach minimizes the need for server-side processing on each user request, resulting in faster loading times and improved overall performance. While the core of a Quartz site is static, the platform also provides mechanisms for incorporating client-side JavaScript to add dynamic and interactive elements to the user experience.10 These JavaScript files are bundled during the build process and included within the generated HTML, allowing for enhanced functionality beyond basic static content.
  To facilitate client-side scripting that interacts with the Document Object Model (DOM) of the page, Quartz dispatches a custom browser event known as "nav".[10] This event is triggered after the initial HTML page has been fully loaded by the browser. Furthermore, if the "enableSPA" (Single Page Application) option is activated in Quartz's configuration, the "nav" event is also fired on any subsequent client-side navigations between pages.[10] This provides a consistent hook for client-side scripts to set up event listeners, manage component state, or perform any other logic that requires access to the page's DOM structure. The "nav" event ensures that these scripts can correctly initialize or refresh their functionality whenever a new page is displayed, regardless of whether it's the initial load or a client-side transition.
  Insight: For the user's objective of embedding NodeBB comments into a Quartz site, client-side JavaScript will likely play a crucial role. This JavaScript can be used to communicate with the NodeBB server, retrieve the relevant comments for a specific page, and dynamically inject them into the page's DOM. The "nav" event dispatched by Quartz can serve as an ideal trigger to ensure that this comment embedding process occurs whenever a page is loaded or navigated to, providing a dynamic commenting experience within the static site.
- Page Metadata (Frontmatter):
  Quartz leverages the concept of frontmatter, a section at the beginning of Markdown files written in YAML format, to allow authors to specify metadata associated with each page.11 This metadata can include various properties such as title for the page title, description for SEO purposes and link previews, permalink to define a custom URL for the page, aliases for alternative names, tags for categorization, draft to control

publishing status, and date to indicate the publication date.12 This information is not only used by Quartz during the site generation process to handle routing, theming, and organization but can also be accessed by client-side scripts running on the generated website.14

Insight: The frontmatter capability of Quartz provides a valuable mechanism for linking static pages to dynamic content from NodeBB. For instance, a unique identifier for each Quartz page, such as its slug or a custom ID, could be stored as a frontmatter property. This identifier could then be used by client-side JavaScript to query the NodeBB API and retrieve the specific comments associated with that particular page, ensuring that the correct discussions are displayed in the embedded comment section.

3. Exploring NodeBB and its Comment Plugin

- NodeBB Forum Software:

  NodeBB is a modern and efficient discussion forum platform built on the Node.js runtime environment, designed to facilitate real-time communication and community building on the web.1 It offers support for a variety of database systems, including the key-value store Redis, the document database MongoDB, and the relational database PostgreSQL, providing flexibility in deployment and data management.8 Key features of NodeBB include real-time streaming discussions, ensuring that users see new posts and replies instantly; a responsive design that adapts seamlessly to different screen sizes and devices; a robust user management system that supports registration, login, and user profiles; the organization of discussions into categories and topics; the use of tags for content categorization and discovery; comprehensive search functionality to help users find relevant information; a real-time notification system to alert users of new activity; and a suite of moderation tools to help maintain a healthy and productive community.5 Furthermore, NodeBB's architecture is highly extensible through its vibrant plugin system, allowing developers to add a wide range of additional features and integrations to their forums.5 The platform also provides a well-documented RESTful API, which enables seamless integration with other web applications and services.6

  Insight: NodeBB's comprehensive feature set, its foundation in Node.js, and the availability of a RESTful API make it a strong candidate for providing the desired comment functionality to a static website built with Quartz. The real-time capabilities of NodeBB could translate to a dynamic and engaging commenting experience for users on the Quartz site.

- NodeBB Blog Comments Plugin:

  The nodebb-plugin-blog-comments is an officially maintained plugin for NodeBB that is specifically designed to allow the forum software to function as a dedicated commenting engine for blog platforms.16 This plugin offers out-of-the-box integration support for popular blogging platforms such as Ghost and WordPress, providing detailed installation and usage instructions for these systems.17 Additionally, there is a separate repository dedicated to supporting integration with the PencilBlue blogging platform.17

  The installation process for the plugin is straightforward, typically involving the use of npm (Node Package Manager) to install the plugin into the NodeBB installation

directory. Once installed, the plugin needs to be activated through the NodeBB Admin Control Panel (ACP).[17] A crucial configuration step involves selecting the specific category within the NodeBB forum where comments originating from the blog will be published and managed.[17] The plugin operates by requiring the embedding of a small JavaScript snippet into the template of the blog posts where comments are desired.[17] This JavaScript code gathers essential information about the blog post, such as its unique ID, title, and the main content (often in Markdown format), and sends this data to the configured NodeBB forum.[17]
Insight: The nodebb-plugin-blog-comments plugin appears to be the most direct and readily available solution for integrating NodeBB's commenting capabilities into an external website. While its primary focus is on established blogging platforms, the plugin also features a "General Use" section within its documentation [69], which suggests that it can be adapted for use with other types of websites, including static sites generated by tools like Quartz. The plugin's method of embedding comments through JavaScript aligns well with the client-side nature of static site integration.

- NodeBB API for Custom Integration:
  NodeBB features a comprehensive RESTful API that allows developers to interact with the forum software programmatically, enabling a wide range of custom integrations.6 This API is divided into two main parts: the Read API, which provides endpoints for fetching data such as forum settings, categories, topics, posts, and user information, and the Write API, which includes endpoints for performing actions like creating new topics, posts, users, and managing user relationships.19
  Authentication to the NodeBB API is primarily handled through the use of bearer tokens. These tokens can be generated within the NodeBB Admin Control Panel and are included in the headers of API requests to authorize the actions being performed.[19] The availability of both Read and Write APIs offers developers a high degree of flexibility in how they can integrate NodeBB with other applications.
  Insight: While the nodebb-plugin-blog-comments plugin provides a more streamlined approach for common blog commenting use cases, the NodeBB API offers the potential for a much more tailored and deeply integrated solution with a Quartz static website. By directly interacting with the API using custom JavaScript code on the Quartz site, developers could have fine-grained control over how comments are fetched, displayed, and submitted, potentially leading to a more seamless user experience and greater flexibility in terms of styling and authentication.

4. Integrating NodeBB Comments into Quartz: Step-by-Step Guide

- **Method 1: Using the NodeBB Blog Comments Plugin (General Use Script):**
  - **Step 1:** Begin by installing the nodebb-plugin-blog-comments plugin on your NodeBB instance. This can be done using npm with the command npm install nodebb-plugin-blog-comments. After the installation is complete, activate the plugin through the NodeBB Admin Control Panel (ACP) and navigate to the Blog Comments section to configure the category within your forum where you want blog comments to be posted.[17]

    - **Step 2:** Within your Quartz project, identify the specific component or layout template file where you intend to embed the comment section. This will likely be a .tsx file located within the quartz/components/pages directory or a custom component you have created.[14]
    - **Step 3:** Locate the general-use JavaScript embed code provided by the nodebb-plugin-blog-comments. This snippet is typically found in the plugin's README.md file on GitHub or within the plugin's documentation on platforms like npm.[21] The code usually involves a <script> tag that initializes a JavaScript object, often named nbb.
    - **Step 4:** Adapt the JavaScript snippet within your Quartz component to dynamically set the necessary variables. The most crucial variable is nbb.articleID, which should be a unique identifier for each page on your Quartz site. A common approach is to use the page's slug, which is generally accessible within Quartz components. Alternatively, you can use a custom identifier that you might have defined in the frontmatter of your Markdown files.[10] Depending on the plugin's specific implementation, you might also need to set nbb.articleTitle and nbb.articleContent to the title and content of your Quartz page.
        - **Insight:** While Quartz utilizes JSX for its component templating, it handles dynamic behavior and user interactions primarily through standard JavaScript event listeners, rather than React-specific hooks like useEffect.[14] You can use the "nav" event listener, which Quartz dispatches after each page load, to execute the JavaScript code responsible for setting up the comment embedding on the current page.[14]
    - **Step 5:** Ensure that the nbb.url variable in the JavaScript snippet is correctly configured to point to the base URL of your NodeBB forum. This is essential for the embedded comment widget to communicate with your forum instance.
    - **Step 6:** Place the anchor tag <a id="nodebb-comments"></a> within your Quartz component at the precise location where you want the comment section to be rendered on the page. This tag serves as a target for the NodeBB comment widget to be inserted into.[17]
    - **Step 7:** Build your Quartz site using the command npx quartz build. This will generate the static HTML, CSS, and JavaScript files for your website, including the embedded comment section. Finally, deploy the generated files to your chosen hosting provider.[1]
    - **Insight:** The articleID plays a vital role in ensuring that comments are correctly associated with the specific page on your Quartz site. Utilizing the page's slug as the articleID is a common and effective method for static site generators to maintain a unique reference for each content page.
- **Method 2: Embedding NodeBB via an Iframe:**
    - **Step 1:** In the NodeBB Admin Control Panel, navigate to Settings > Advanced > Headers. Locate the Content-Security-Policy header and within the frame-ancestors directive, add the domain of your Quartz website to the list of

allowed sources. This step is crucial for permitting your Quartz site to embed content from your NodeBB forum within an <iframe>.[23] You might also need to adjust the Access-Control-Allow-Origin header in the same settings section.
    - **Step 2:** In the specific component or layout file within your Quartz project where you want the forum to appear, embed an <iframe> HTML element.[26]
    - **Step 3:** Set the src attribute of the <iframe> to the URL of the specific NodeBB topic or category that you wish to display as the comment section for the current Quartz page. This will require a system for linking each Quartz page to a particular discussion thread or category in your NodeBB forum. One way to achieve this is by using a consistent naming convention for topics that mirrors your Quartz page slugs. Alternatively, you could add a custom field to the frontmatter of your Quartz pages to store the corresponding NodeBB topic ID or URL, or maintain a separate mapping table that correlates Quartz pages with their respective NodeBB comment threads.
    - **Step 4:** Adjust the width and height attributes of the <iframe> element to ensure it is displayed correctly within the layout of your Quartz site. For better responsiveness across different devices, consider using CSS to manage the iframe's dimensions.[26]
    - **Insight:** While embedding NodeBB using an iframe might seem like a relatively straightforward approach for initial setup, it can present several limitations. Maintaining a consistent visual style between the Quartz site and the content within the iframe can be challenging. Additionally, iframes can have potential negative impacts on SEO [27], and the user experience might be less seamless due to the isolated nature of the embedded content.
- **Method 3: Custom Integration using NodeBB's API (Conceptual):**
    - **Step 1:** Begin by thoroughly reviewing the documentation for both the Quartz plugin system and the NodeBB REST API.[10] Consider the possibility of developing a custom plugin for Quartz or utilizing JavaScript within your Quartz components to directly interact with the various endpoints provided by the NodeBB API.
    - **Step 2:** Implement a system for user authentication that spans both your Quartz site and your NodeBB forum. This could involve leveraging NodeBB's Write API to create and authenticate users directly from your Quartz site's frontend, or by implementing a Single Sign-On (SSO) solution using plugins like nodebb-plugin-session-sharing or nodebb-plugin-sso-oauth, which can synchronize user sessions across multiple applications.[31]
    - **Step 3:** Use JavaScript code within your Quartz components to fetch comments for the currently viewed page by making requests to NodeBB's Read API. You will likely need to filter these requests based on a unique identifier that links the specific Quartz page to the corresponding discussion thread within your NodeBB forum.
    - **Step 4:** Implement a comment submission form directly within your Quartz site's template. When a user submits a comment, the form would use JavaScript to

send the comment data to NodeBB's Write API, creating a new post within the designated topic in your NodeBB forum.

  - **Insight:** This approach offers the highest level of flexibility and customization for integrating NodeBB's commenting functionality into your Quartz site. However, it also requires a significant development effort and a deep understanding of the internal workings and APIs of both platforms. Successfully implementing this method could result in the most seamless integration in terms of visual appearance, user authentication, and data management.

5. **Potential Challenges and Considerations**
   - **Cross-Origin Resource Sharing (CORS):** If your Quartz website and NodeBB forum are hosted on different domains or ports, you will likely encounter issues with Cross-Origin Resource Sharing (CORS) when attempting to embed content or make API requests between the two. This is a security feature implemented by web browsers to prevent malicious websites from making unauthorized requests to other domains. To overcome this, you will need to configure the CORS headers on your NodeBB server to explicitly allow requests originating from your Quartz site's domain.[23] This configuration typically involves specifying the allowed origins in the server's settings.
   - **Styling and Theme Consistency:** Achieving a visually consistent user experience between your static Quartz website and the dynamically embedded NodeBB comments can be a significant challenge. When using iframes, the styling of the content within the iframe is often isolated from the parent page, potentially leading to visual discrepancies. While the nodebb-plugin-blog-comments plugin might provide some basic default styling for the embedded comments, you may need to implement custom CSS within your Quartz theme or potentially within NodeBB's theme to override or supplement these styles and ensure a seamless integration.[5]
   - **Search Engine Optimization (SEO):** Content that is embedded into a website using client-side JavaScript might not be indexed by search engines as effectively as content that is directly rendered within the initial HTML source of the page. If ensuring that your comments are discoverable by search engines is a primary concern, you might need to explore alternative commenting solutions that offer server-side rendering of comments or ensure that the JavaScript embedding method you choose is implemented in a way that allows search engine crawlers to access and index the comment content.[27]
   - **Performance Impact:** Embedding external resources, particularly through the use of JavaScript or iframes, can potentially increase the loading time of your static website. This can negatively impact the user experience and potentially affect your site's performance metrics. To mitigate this, consider implementing lazy loading for the comment section. This technique involves delaying the loading of the comment section until the user is about to scroll to that part of the page or interacts with a specific element that triggers the loading of the comments.[43]
   - **User Authentication and Session Management:** If your Quartz website has its own system for managing user authentication (i.e., user logins), you will need to carefully consider how to handle user logins and sessions for the embedded NodeBB comments

to provide a cohesive user experience. This might involve exploring NodeBB's Single Sign-On (SSO) capabilities, which allow users to log in once and access multiple applications, or utilizing a session-sharing plugin that can synchronize user authentication status between your Quartz site and the NodeBB forum.[31]

- **Linking Quartz Pages to NodeBB Topics:** To ensure that the correct comments are displayed on each page of your Quartz website, you will need to establish a clear and maintainable method for associating each static page with a corresponding discussion thread or category within your NodeBB forum. This could involve adopting a consistent naming convention for topics within NodeBB that mirrors the slugs or file names of your Quartz pages, using a custom field within the frontmatter of your Quartz pages to store the ID or URL of the relevant NodeBB topic, or maintaining a separate mapping table that explicitly correlates each Quartz page with its associated NodeBB comment thread.
- **Plugin Compatibility and Maintenance:** When opting to use plugins like nodebb-plugin-blog-comments to facilitate the integration, it is essential to verify that the plugin is compatible with the specific version of NodeBB you have installed. Furthermore, you should be prepared to monitor and maintain the integration over time, as both Quartz and NodeBB, along with their respective plugins, will likely receive updates that might require adjustments to your setup.[49]

6. **Exploring Alternative Commenting Solutions for Quartz**

- **Third-Party Commenting Systems:**
  - **Disqus:** One of the most widely recognized hosted commenting platforms, Disqus offers a free tier with a range of features, including moderation tools, social media integration, and analytics. However, the free tier typically includes advertisements, and there have been discussions regarding its privacy practices.
  - **Commento:** A fast, privacy-focused, and open-source commenting platform that offers both a paid hosted service and the option for self-hosting. It is known for its lightweight design and provides tools for moderation and spam filtering. Commento also allows users to import existing comments from Disqus.
  - **Hyvor Talk:** Marketed as a privacy-first alternative to Disqus, Hyvor Talk offers a free plan and a range of features, including customization options and support for multiple languages. It also provides tools for moderation and spam detection.[51]
  - **CommentBox.io:** This hosted commenting platform provides a clean and minimalist comment box design. It offers a free tier for personal use with limitations on the number of comments and paid plans for more extensive usage.[51]
  - **Insight:** These third-party systems provide a relatively easy way to add commenting functionality to a static site by embedding JavaScript snippets. However, they involve relying on external services, which might raise concerns about privacy, data control, and long-term availability.
- **Self-Hosted Open-Source Platforms:**
  - **Commento (Self-Hosted):** Requires server infrastructure capable of running PostgreSQL.[70]
  - **Schnack!:** A lightweight comment system built with Node.js, which can be

    self-hosted and offers support for various authentication methods.[52]
    - **Isso:** A lightweight commenting server written in Python and utilizing SQLite for its database. It is known for its simplicity and ease of embedding into web pages.[51]
    - **Remark42:** An open-source and self-hosted comment system written in Go, designed with a focus on simplicity and reliability. It can be easily deployed using Docker.[52]
    - **Cactus Comments:** A unique option that uses the Matrix protocol, allowing for a federated and potentially more private commenting experience. It can be self-hosted and integrated into both static and dynamic sites.[51]
    - **Insight:** Self-hosting provides greater control over comment data and privacy, as the data is stored on your own server. However, it requires a certain level of technical expertise to set up, configure, and maintain the server infrastructure and the commenting software.
- **Using GitHub Issues or Discussions:**
    - **Utteranc.es:** A free and open-source commenting widget that utilizes GitHub Issues as a data store for comments. It is a lightweight option that requires users to have a GitHub account to participate in the discussions.[52]
    - **Giscus:** Another free and open-source comment system that is powered by GitHub Discussions. It offers a more feature-rich experience compared to Utteranc.es, including support for reactions, theming, and automatic topic creation.[52]
    - **Insight:** These GitHub-based solutions are particularly well-suited for websites that cater to a developer audience, as they seamlessly integrate with the GitHub platform and leverage its existing user base and features.
- **Webmentions:** A more decentralized approach to handling comments and interactions on the web. Webmentions allow your website to receive notifications when its content is linked to on other sites. This typically requires integration with a third-party service like webmention.io or brid.gy to facilitate the sending and receiving of these mentions.[63]

7. Recommendations and Best Practices

- **Recommended Approach:** Based on the research conducted, the most direct and potentially efficient method for integrating NodeBB comments into a Quartz static website is to utilize the nodebb-plugin-blog-comments plugin and embed it using the general-purpose JavaScript snippet provided by the plugin. This approach leverages a pre-existing solution designed for embedding NodeBB comments and has the potential to be adapted for use with Quartz.
- **Step-by-Step Instructions:** (Detailed in the expanded outline under "Integrating NodeBB Comments into Quartz: Step-by-Step Guide - Method 1").
- **Optimizing Performance:** To ensure that the addition of a comment section does not negatively impact the loading speed of your Quartz site, it is highly recommended to implement lazy loading for the comment section. This can be achieved by using the loading="lazy" attribute on the <script> tag that loads the NodeBB comment plugin, or

by employing JavaScript-based lazy loading techniques that delay the loading of the comment section until the user is about to view it.[43]

- **SEO Considerations:** While content embedded via JavaScript can be indexed by search engines, it is advisable to monitor your website's performance in search results using tools like Google Search Console after implementing the comment section. If SEO for your comments is a primary goal, you might consider exploring alternative commenting solutions that offer server-side rendering or more direct HTML output of the comment content to ensure better indexability.[40]
- **Handling Challenges:**
  - To address potential CORS issues that might arise from embedding content from a different domain, ensure that you configure the appropriate headers on your NodeBB server to explicitly allow requests from your Quartz site's domain.
  - Achieving visual consistency between your static Quartz website and the embedded NodeBB comments might require some effort. You can use custom CSS within your Quartz theme to target and style the elements injected by the NodeBB plugin to match your site's overall design.
  - If you require user authentication for commenting, explore the Single Sign-On (SSO) capabilities offered by NodeBB or consider using its REST API to integrate with an existing authentication system on your Quartz site for a more seamless user experience.
  - Establish a clear and maintainable method for linking each page on your Quartz site to the corresponding comment thread or category in your NodeBB forum. Using the page's slug as a basis for the NodeBB topic title or creating a custom field in the Quartz page's frontmatter to store the relevant NodeBB topic ID are viable strategies.

8. Conclusion

Integrating NodeBB comments into a Quartz static website is a feasible endeavor, with the nodebb-plugin-blog-comments offering a practical starting point. By following the recommended steps and carefully considering the potential challenges related to CORS, styling, SEO, and performance, you can enhance your static site with dynamic and engaging comment functionality. While the nodebb-plugin-blog-comments plugin provides a relatively straightforward method, remember that a variety of alternative commenting systems are also available, each with its own strengths and weaknesses. Ultimately, the best approach will depend on your specific requirements, technical expertise, and priorities for user experience, performance, and SEO. Adding a robust commenting system like NodeBB can significantly enhance user engagement and foster a sense of community around the content on your Quartz static website.

Key Tables to Include:

- **Section 6: Exploring Alternative Commenting Solutions for Quartz**
  - **Table Title:** Comparison of Commenting Systems for Static Sites
  - **Columns:** Name, Hosting Model, Open Source, Free Tier/Basic Cost, Key Features, Integration Method, SEO Considerations
  - **Data:**

Name	Hosting Model	Open Source	Free Tier/Basic Cost	Key Features	Integration Method	SEO Considerations
Disqus	Hosted	No	Free (with ads)/Paid	Moderation, social login, analytics, import/export	JavaScript embed	Can be indexed by Google; offers options for local caching and emitting comments into the parent page, but might require specific configuration.
Commento	Hosted & Self-Hosted	Yes	Paid/Self-Hosted	Privacy-focused, moderation, spam filtering, Markdown support, Disqus import	JavaScript embed	Generally good for SEO as it aims to be lightweight and fast-loading.
Hyvor Talk	Hosted	No	Free/Paid	Privacy-first, customizable, moderation, multi-language support, SSO	JavaScript embed	Designed to be SEO-friendly.
Utteranc.es	GitHub Issues (Self-Hosted)	Yes	Free	Lightweight, uses GitHub Issues for storage, Markdown support	JavaScript embed	Comments are stored as GitHub Issues, which are publicly accessible and indexed by search engines.

Giscus	GitHub Discussions (Self-	Yes	Free	Uses GitHub Discussions, reactions, theming, automatic topic creation	JavaScript embed	Leverages GitHub Discussions, which are indexed by search engines.
Schnack!	Self-Hosted	Yes	Free	Simple, Node.js-based, 3rd party auth support	JavaScript embed	SEO considerations depend on the implementation and how the comments are rendered.
Isso	Self-Hosted	Yes	Free	Lightweight, Python-based, SQLite backend, Markdown support, moderator tools	JavaScript embed	SEO considerations depend on the implementation.
Remark42	Self-Hosted	Yes	Free	Go-based, Docker deployment, OAuth logins, Disqus/WordPress migration	JavaScript embed	SEO considerations depend on the implementation.
Cactus Comments	Self-Hosted	Yes	Free	Matrix protocol, federated architecture, supports multiple platforms	JavaScript embed	SEO considerations depend on the implementation.
Webmentions	Decentralized (Service	Yes	Free/Service Cost	Receives notifications	Link in <head>, API	Mentions themselves

	Req.)			of links, likes, etc., from other websites	usage	might not be traditional comments, but the content on the linking page would be indexed. Requires careful implementation for comment-like functionality.

*   **Reasoning:** This table provides a structured overview of the most relevant alternatives, allowing the user to quickly compare options based on their specific needs and priorities, such as hosting requirements, cost, features, integration complexity, and potential impact on SEO, aiding them in making an informed decision should the direct NodeBB integration not be ideal for their specific needs.

Works cited

1. Quartz - Fork My Brain, accessed April 18, 2025, https://notes.nicolevanderhoeven.com/Quartz
2. Quartz - Publish Obsidian Vault - Data Engineering Blog, accessed April 18, 2025, https://www.ssp.sh/brain/quartz-publish-obsidian-vault/
3. Welcome to Quartz 4, accessed April 18, 2025, https://quartz.jzhao.xyz/
4. jackyzha0/quartz: a fast, batteries-included static-site generator that transforms Markdown content into fully functional websites - GitHub, accessed April 18, 2025, https://github.com/jackyzha0/quartz
5. NodeBB Features & Capabilities | GetApp, accessed April 18, 2025, https://www.getapp.com/website-ecommerce-software/a/nodebb/features/
6. Product | NodeBB | Modern Forum Software, accessed April 18, 2025, https://nodebb.org/product
7. Home - NodeBB - Modern Community Forum Software, accessed April 18, 2025, https://nodebb.org/
8. NodeBB/NodeBB: Node.js based forum software built for ... - GitHub, accessed April 18, 2025, https://github.com/NodeBB/NodeBB

9. Empowering Developer Communities with NodeBB: A Modern Forum Platform - Habilelabs, accessed April 18, 2025, https://www.habilelabs.io/blog/empowering-developer-communities-with-nodebb
10. Architecture - Quartz 4, accessed April 18, 2025, https://quartz.jzhao.xyz/advanced/architecture
11. Building a static website with Quartz, Markdown, and Cloudflare Pages - Christopher Klint, accessed April 18, 2025, https://christopherklint.com/blog/building-a-static-website-with-quartz-markdown-cloudflare-pages
12. NodeBB as a Site? Using it to build my Tech Blog maybe. : r/fediverse - Reddit, accessed April 18, 2025, https://www.reddit.com/r/fediverse/comments/1iuwovp/nodebb_as_a_site_using_it_to_build_my_tech_blog/
13. Authoring Content - Quartz 4, accessed April 18, 2025, https://quartz.jzhao.xyz/authoring-content
14. Creating your own Quartz components, accessed April 18, 2025, https://quartz.jzhao.xyz/advanced/creating-components
15. NodeBB Community: Home, accessed April 18, 2025, https://community.nodebb.org/
16. NodeBB Plugins, accessed April 18, 2025, https://community.nodebb.org/category/7/nodebb-plugins
17. nodebb-plugin-blog-comments - NPM, accessed April 18, 2025, https://www.npmjs.com/package/nodebb-plugin-blog-comments
18. [nodebb-plugin-blog-comments] Blog Commenting Engine (Ghost, Wordpress widget), accessed April 18, 2025, https://community.nodebb.org/topic/810/nodebb-plugin-blog-comments-blog-commenting-engine-ghost-wordpress-widget
19. NodeBB Read API (3.0.0) - ReDoc, accessed April 18, 2025, https://docs.nodebb.org/api/read/
20. A RESTful JSON-speaking API allowing you to write things to NodeBB - GitHub, accessed April 18, 2025, https://github.com/NodeBB/nodebb-plugin-write-api
21. revir/nodebb-plugin-blog-comments2: Lets NodeBB act as a comments engine/widget for your blog - GitHub, accessed April 18, 2025, https://github.com/revir/nodebb-plugin-blog-comments2
22. Building digital gardens with QUARTZ – Static Feedback #9 - YouTube, accessed April 18, 2025, https://www.youtube.com/watch?v=YCvV7Izqggc
23. How to integrate nodebb in iframe in reactjs - Stack Overflow, accessed April 18, 2025, https://stackoverflow.com/questions/58407991/how-to-integrate-nodebb-in-iframe-in-reactjs
24. How can I load NodeBB in an iframe?, accessed April 18, 2025, https://community.nodebb.org/topic/1678/how-can-i-load-nodebb-in-an-iframe
25. Allow forum to be embedded on multiple domains? - NodeBB Community, accessed April 18, 2025,

https://community.nodebb.org/topic/15788/allow-forum-to-be-embedded-on-multiple-domains
26. Using iFrames on your static website, accessed April 18, 2025, https://simplystatic.com/tutorials/iframe-static-website/
27. Embed your Forum - The Success Community - Higher Logic Vanilla, accessed April 18, 2025, https://success.vanillaforums.com/kb/articles/67-embed-your-forum
28. Embed web page within post | NodeBB Community, accessed April 18, 2025, https://community.nodebb.org/topic/3858/embed-web-page-within-post
29. Documentation on how to use NodeBB available funtions and methods, accessed April 18, 2025, https://community.nodebb.org/topic/16140/documentation-on-how-to-use-nodebb-available-funtions-and-methods
30. ReDoc, accessed April 18, 2025, https://docs.nodebb.org/api/
31. Can nodebb be embedded into an existing website?, accessed April 18, 2025, https://community.nodebb.org/topic/15316/can-nodebb-be-embedded-into-an-existing-website
32. nodebb-plugin-session-sharing - NPM, accessed April 18, 2025, https://www.npmjs.com/package/nodebb-plugin-session-sharing?activeTab=readme
33. What's the best strategy to integrate nodeBB with another account system?, accessed April 18, 2025, https://community.nodebb.org/topic/14916/what-s-the-best-strategy-to-integrate-nodebb-with-another-account-system
34. How I can Integrate NodeBB into my website, accessed April 18, 2025, https://community.nodebb.org/topic/10222/how-i-can-integrate-nodebb-into-my-website
35. Headless Integration with Nodebb, accessed April 18, 2025, https://community.nodebb.org/topic/16854/headless-integration-with-nodebb
36. Embedded comments in HTML page - SAMEORIGIN and 403 issues - Discourse Meta, accessed April 18, 2025, https://meta.discourse.org/t/embedded-comments-in-html-page-sameorigin-and-403-issues/50647
37. How to style embedded comments? - Support - Discourse Meta, accessed April 18, 2025, https://meta.discourse.org/t/how-to-style-embedded-comments/25348
38. Big Newb. Need Help With CSS Customization | NodeBB Community, accessed April 18, 2025, https://community.nodebb.org/topic/12301/big-newb-need-help-with-css-customization
39. How to Add a Forum to Your Website: Easy Setup Explained | Bettermode, accessed April 18, 2025, https://bettermode.com/blog/how-to-add-a-forum-to-your-website
40. Add comments to 'static' pages without Javascript so they can be index by search engines?, accessed April 18, 2025, https://webmasters.stackexchange.com/questions/28611/add-comments-to-stati

c-pages-without-javascript-so-they-can-be-index-by-search

41. JavaScript SEO: How Google Crawls and Indexes JavaScript Websites - iPullRank, accessed April 18, 2025, https://ipullrank.com/javascript-seo-how-google-crawls-and-indexes-javascript-websites
42. Understand JavaScript SEO Basics | Google Search Central | Documentation, accessed April 18, 2025, https://developers.google.com/search/docs/crawling-indexing/javascript/javascript-seo-basics
43. Comments on static sites? : r/webdev - Reddit, accessed April 18, 2025, https://www.reddit.com/r/webdev/comments/qwaekw/comments_on_static_sites/
44. FediForum hot take - NodeBB Community, accessed April 18, 2025, https://community.nodebb.org/topic/18285/fediforum-hot-take
45. javascript - implementing lazyload using html comments - Stack Overflow, accessed April 18, 2025, https://stackoverflow.com/questions/30744731/implementing-lazyload-using-html-comments
46. How to Lazy Load WordPress Images, Video, Comments + iframes - Themeisle, accessed April 18, 2025, https://themeisle.com/blog/lazy-load-wordpress/
47. Best practices for using third-party embeds | Articles - web.dev, accessed April 18, 2025, https://web.dev/articles/embed-best-practices
48. Lazy Load For Comments Tutorial 2020 - WordPress - YouTube, accessed April 18, 2025, https://www.youtube.com/watch?v=LR23IHXNEWc
49. NodeBB Blog Comments Plugin Ghost.js Not Found - Cloudron Forum, accessed April 18, 2025, https://forum.cloudron.io/topic/6973/nodebb-blog-comments-plugin-ghost-js-not-found
50. [nodebb-plugin-blog-comments] Need help | NodeBB Community, accessed April 18, 2025, https://community.nodebb.org/topic/12614/nodebb-plugin-blog-comments-need-help
51. 6 Tools to Use Comments on a Static Site, accessed April 18, 2025, https://simplystatic.com/tutorials/comments-on-a-static-site/
52. The ultimate guide to comments for static sites - Shifter, accessed April 18, 2025, https://getshifter.io/static-site-comments/
53. Top 11 Disqus Alternatives in 2025 - HYVOR, accessed April 18, 2025, https://hyvor.com/blog/disqus-alternatives
54. What is the recommended way to add comment sections to your Jekyll blog?, accessed April 18, 2025, https://talk.jekyllrb.com/t/what-is-the-recommended-way-to-add-comment-sections-to-your-jekyll-blog/3330
55. Comment systems for static websites - Dan MacKinlay, accessed April 18, 2025, https://danmackinlay.name/notebook/static_site_comments.html
56. How to add a free, GitHub's issues-based commenting system to a Gatsby static site, accessed April 18, 2025, https://antran.app/2022/gatsby_utterances/

57. LeeHolmes/statique: Simple Self-Hosted Comments for Static Websites - GitHub, accessed April 18, 2025, https://github.com/LeeHolmes/statique
58. Adding comments to a Jekyll site with Staticman - Made Mistakes, accessed April 18, 2025, https://mademistakes.com/mastering-jekyll/static-comments/
59. Add comments with utterances - Franklin.jl, accessed April 18, 2025, https://franklinjl.org/extras/utterances/
60. How to add Comments section for free on your website? - DEV Community, accessed April 18, 2025, https://dev.to/yashkapure06/how-to-add-comments-section-for-free-on-your-website-49n5
61. Replace the Disqus Commenting System on Your Blog, accessed April 18, 2025, https://www.cesarsotovalero.net/blog/replace-disqus-with-a-better-alternative.html
62. How to embed GitHub "Discussions" thread within your website, accessed April 18, 2025, https://webmasters.stackexchange.com/questions/138530/how-to-embed-github-discussions-thread-within-your-website
63. Static Site Comments: A Jamstack How-To - Snipcart, accessed April 18, 2025, https://snipcart.com/blog/jamstack-static-site-comments
64. Adding webmentions to your static blog - Jan Monschke, accessed April 18, 2025, https://janmonschke.com/adding-webmentions-to-your-static-blog/
65. Combining Hugo with phpBB/Discourse/nodeBB to provide comments? : r/Jekyll - Reddit, accessed April 18, 2025, https://www.reddit.com/r/Jekyll/comments/jfy9lr/combining_hugo_with_phpbbdiscoursenodebb_to/
66. Open-source self-hosted comments systems for static websites | Hacker News, accessed April 18, 2025, https://news.ycombinator.com/item?id=24676152
67. 🌱 adding a comments section to a static site | Let's get Person-al, accessed April 18, 2025, https://person-al.github.io/%F0%9F%8C%B1/2024/01/14/adding-a-comments-section-to-a-static-site.html
68. Alternative to Disqus Needed More Than Ever - Page 4 - support - HUGO, accessed April 18, 2025, https://discourse.gohugo.io/t/alternative-to-disqus-needed-more-than-ever/5516?page=4
69. Using NodeBB as a commenting system (static site), accessed April 18, 2025, https://community.nodebb.org/topic/12598/using-nodebb-as-a-commenting-system-static-site
70. Tutorial: How to Add Self-hosted Comments to Your Static Site - Blogging for Devs, accessed April 18, 2025, https://bloggingfordevs.com/static-site-comments/