Roblox Security Best Practices

Roblox · https://namazustudios.com/docs/add-ons/roblox/roblox-security-best-practices/

Pre-Release

Note: The Namazu Elements Roblox Kit is currently in pre-release. To access the Roblox Support in Namazu Elements please reach out and request access using our Contact Us Form.

When using the Namazu Elements Roblox Kit, it's crucial to follow security best practices to protect your game and players. Here are key guidelines:

- **Server-Side Only:** Always use **server-side scripts** to interact with Namazu Elements services. Never call the Roblox Kit endpoints directly from a LocalScript or client-side, as that could expose your secret or allow tampering. Keep all HTTP requests confined to Roblox's Script/ModuleScript running on the server (e.g. in ServerScriptService).
- **Protect Your Application Secret:** Store your Namazu Application Secret in **Roblox Secrets** (the cloud key management service) and **never hard-code it** in your scripts or expose it to players. Roblox Secrets ensure the key is encrypted and only accessible to the server at runtime. It's also recommended to **rotate** (change) your secret periodically and update it both in Namazu's settings and Roblox Secrets storage. This minimizes the risk if a secret were ever compromised.
- **Safeguard Session Tokens:** Treat the session tokens (`sessionSecret`) as sensitive credentials. Do **not log them to the output** or expose them to players or untrusted sources. These tokens grant access to Namazu services on behalf of a user, so they should be kept confidential. Use them only in authorized server calls and if you temporarily store them (e.g. in a server variable), ensure they can't be read by any client logic. If a player leaves or a server shuts down, you can discard that token.
- **Use HTTPS and Roblox Security Features:** All communication with Namazu should be done over HTTPS (which is enforced by using the provided `*.cloud.namazustudios.com` endpoints). Make sure **HttpService** is enabled in your game settings and use Roblox's built-in security features like **pcall** when making HTTP requests to gracefully handle errors. Always check responses and handle failures (e.g., if authentication fails or matchmaking is unavailable) in your code – the example

code in the kit demonstrates wrapping `HttpService:PostAsync` calls in `pcall` for safety. By following Roblox’s guidelines for web calls and using the Namazu Kit as intended, you maintain a secure environment for your game’s online features.