Matchmaking – Comprehensive Guide

Configuration · https://namazustudios.com/docs/configuration/matchmaking/

Namazu Elements supports matchmaking for online gaming, including one vs one or group matches.

Matchmaking Application Configurations are per-application settings that determine how players are grouped together into matches. Starting with **Namazu Elements 3.4**, all matchmaking functions are handled through the **MultiMatch API**, which supports arbitrarily large player counts and custom matchmaking logic. Each configuration defines its own matchmaking queue, allowing multiple modes (for example, **ranked** and **unranked**) within the same application. Configuration and management occur through the **Namazu Elements CMS**.

This guide combines official documentation with hands-on instructions to explain all key operations:

- **Understanding what a configuration does.**
- **Creating a new matchmaking configuration.**
- **Updating an existing configuration.**
- **Deleting a configuration.**

Key Matchmaking Features

- **Scalable player counts** – Supports matches with arbitrarily large numbers of players, exceeding practical limits for most modern games.
- **Custom matchmaking logic** – Allows developers to define bespoke matchmaking schemes such as **Skill-Based**, **Friend Code**, or other custom lobby systems.
- **Matchmaking Application Configurations** – Centralized definitions for player count, team structure, and matchmaking behavior.
- **CMS integration** – Full setup and configuration can be managed through the **Namazu Elements CMS** interface.

- **Use Default Matchmaker** – When checked, uses the built-in FIFO (first-in-first-out) matchmaker to pair playersnamazustudios

This section describes how to configure and deploy matchmaking applications using the MultiMatch API.

What a Matchmaking Configuration Does

Matchmaking Application Configurations are per-application settings that define a set of **matchmaking rules**. They combine key parameters such as player count, timeout settings, and the matchmaking algorithm to determine how players are grouped into matches.

The algorithm itself is implemented as a **plug-in module**, which reads these values when executing matchmaking.

- The **success callback** field is **deprecated** and will be removed in a future release. The related fields remain as legacy placeholders from earlier versions.
- This configuration allows you to define a **custom system for matching players** based on your game's or application's rules.
- Namazu Elements supports virtually any matchmaking system with only a few lines of backend code.
- A **MultiMatch** created from this configuration pairs players within the same application.
- Each matchmaking configuration represents a **separate matchmaking queue**, enabling multiple concurrent matching modes (e.g., *ranked* or *unranked*).

Matchmaking Configuration Fields

Prerequisites

- id – The unique database-id of the Application Configuration itself
- **Name** – A unique identifier for the configuration. Names must start with a letter or digit and may include underscores. Each configuration's name is used in API calls and in the CMS.

- **Description** – A human-readable description of the configuration (e.g., “Unranked matchmaking configuration for casual games”).
- **Max Profiles** – The maximum number of player profiles (players) allowed per match. The minimum value is 2
- **Linger Seconds** – How long a match remains in the database after it ends (default 300 seconds). This is useful for post-match experiences such as victory screens
- **Timeout Seconds** – The maximum lifetime of a match before it is automatically deleted (default 86 400 seconds, or 24 hours)
- **Use Default Matchmaker** – When checked, uses the built-in FIFO (first-in-first-out) matchmaker to pair players. You can uncheck this and specify a custom **Element Name**, **Service Type** and **Service Name** to implement bespoke matchmaking logic.

This document provides a step-by-step guide on how to update and remove a matchmaking configuration from an application in **Namazu Elements**. Changing its name or adjusting the maximum profiles (player count) affects how matches are formed. Removing a configuration will completely delete it from the application, so be careful when performing deletion.

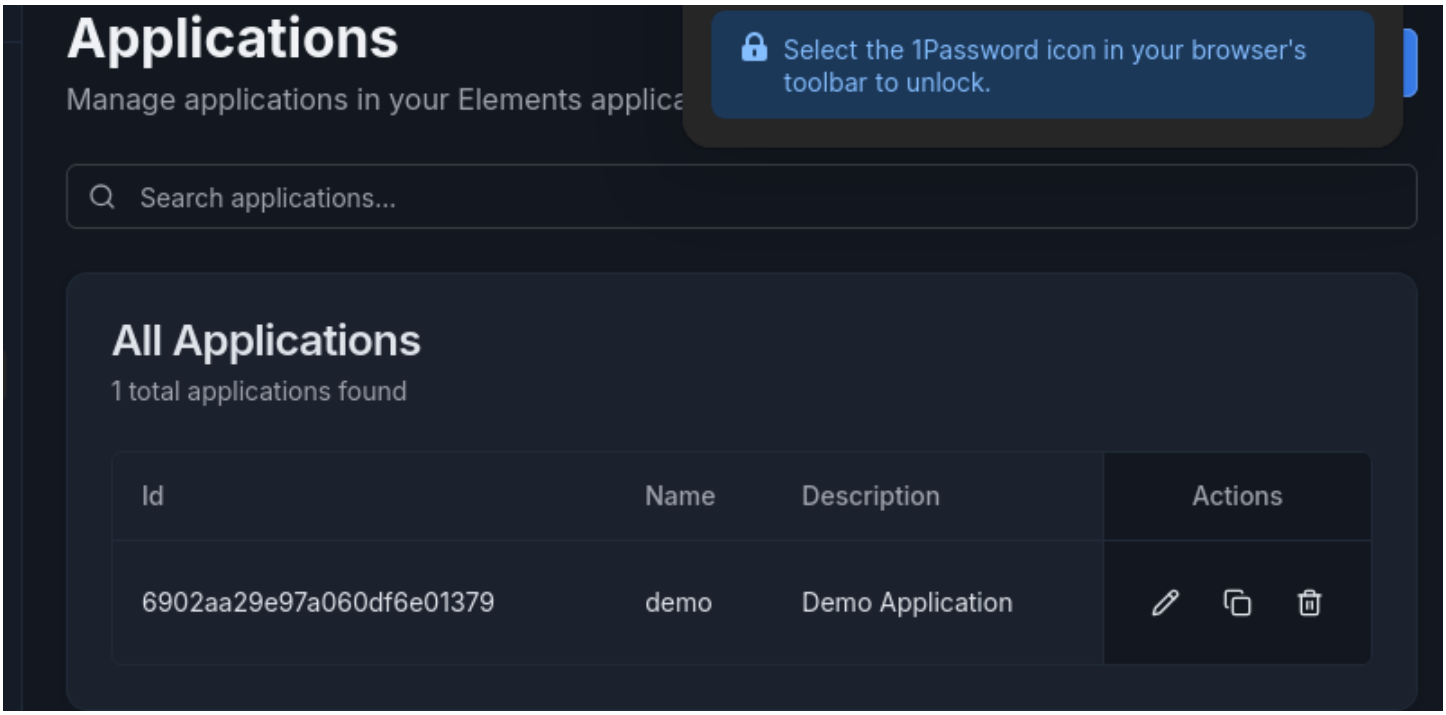
- A Namazu Elements administrator account.
- URL to the admin portal: `https://<your instance>/admin/`.
- Credentials (username and password).
- An existing application with a matchmaking configuration (in this example, **Demo Application** with an initial configuration named **unranked**).

> **Note:** In this guide, all actions are performed while logged into the admin portal and assume you have sufficient permissions. The steps follow a typical workflow of editing an application configuration and then deleting it.

Adding a Matchmaking Configuration

1. Open the Application for Editing

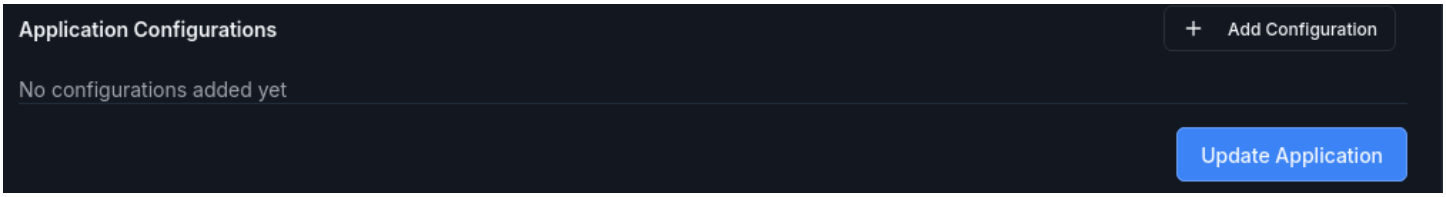
1. From the Namazu Elements dashboard, locate the application you want to modify (for example, *demo*).
2. In the **Actions** column, click the **edit (pencil)** icon to open the **Edit Application** dialog.



Find the Application you Wish to Edit

2. Add a New Configuration

1. In the **Edit Application** dialog, scroll to the **Application Configurations** section.
2. Click **+ Add Configuration** to open the **Application Configuration** form.

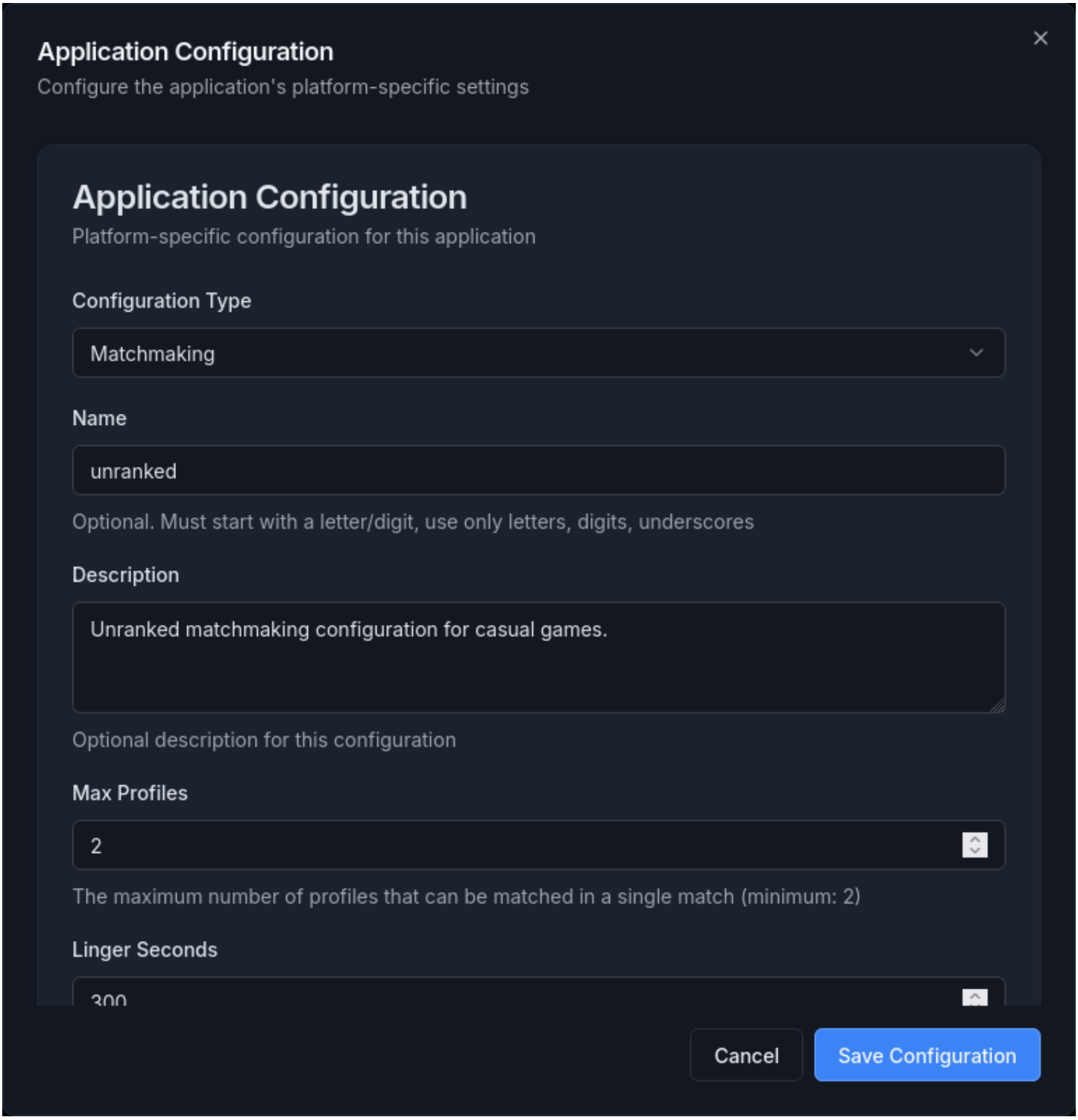


An Empty Application with No Configurations

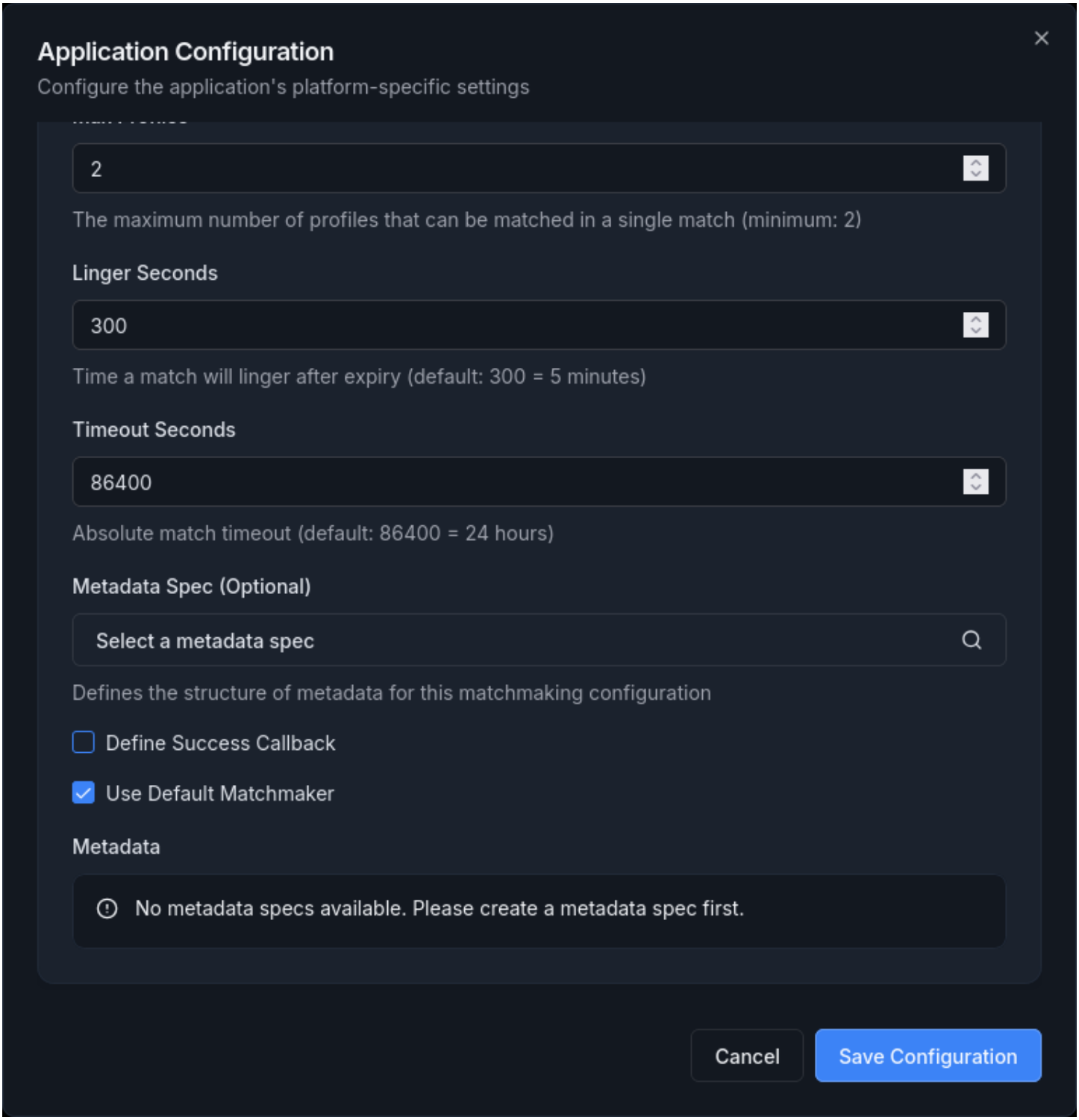
3. Create the Matchmaking Configuration

1. In the **Application** **Configuration** form, ensure **Configuration Type** is set to **Matchmaking**.
2. Enter the following values:
   - **Name:** `unranked`
     (Names must start with a letter or digit and may include underscores.)
   - **Description:** Provide a short description, e.g., *"Unranked matchmaking configuration for casual games."*
   - **Max Profiles:** `2`
     (The maximum number of players per match. The minimum value is 2.)
3. Review and confirm the following additional fields:
   - **Linger Seconds:** `300` (default)
     Defines how long a match remains in the database **after it has ended** before it is permanently deleted.
     This can be used to create **post-match experiences** like victory or celebration screens (e.g., *PUBG* end-of-match sequences).
   - **Timeout Seconds:** `86400` (default)
     Defines the **maximum lifetime** of a match before it is automatically deleted.
     Custom Element code can update this value downstream to extend a match's duration dynamically.
   - **Use Default Matchmaker:** Keep this checked (unless you are adding a custom matchmaker).
     The **default matchmaker** uses the **built-in FIFO (First-In-First-Out)** algorithm to pair players in the order they arrive.
   - **Success Callback:** Leave blank.
     This field is **deprecated** and will be removed in a future release.
4. (Optional) You can customize the matchmaking logic by specifying:
   - **Element** **Name** – the Element responsible for handling matchmaking
   - **Service** **Type** – the backend category managing the logic
   - **Service** **Name** – the specific function or endpoint performing matchmaking

- These values allow you to implement a **custom server-side matchmaking system** within Namazu Elements.



Matchmaking Application Configuration Editor (Upper Half)

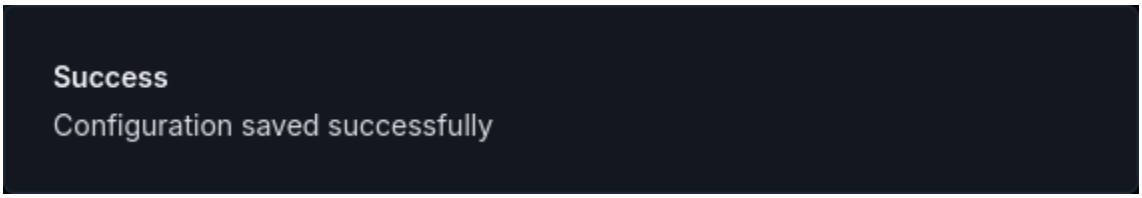


Matchmaking Application Configuration Editor (Lower Half)

4. Save and Verify

1. Click **Save Configuration** at the bottom of the form.

2. A success message (toast) appears confirming that the configuration was saved.
3. The new configuration is now visible in the **Application Configurations** list within the **Edit Application** dialog.

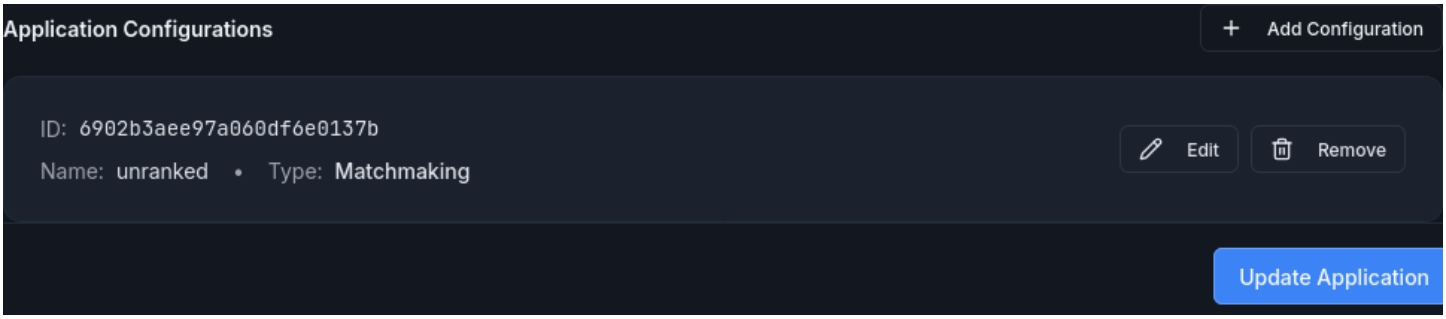


The Success Dialog appears when You Successful Update the Application

---

5. Close the Edit Dialog

1. Click the **X** in the top-right corner of the **Edit Application** dialog to return to the main **Applications** list.
2. Confirm that your new configuration appears under the target application.



The bottom of the Application Editor will Show all Application Configurations

---

Summary

You have created an **unranked** matchmaking configuration for your application using the default FIFO matchmaker.

- **Linger Seconds** controls how long matches persist post-termination, useful for post-game effects.
- **Timeout Seconds** sets the match's maximum lifespan, adjustable via Element code.
- **Success Callback** is deprecated and should remain unused.

- **Default Matchmaker** leverages Namazu's FIFO algorithm but can be replaced with a custom implementation via Element, Service Type, and Service Name.
- Each configuration corresponds to its own **matchmaking queue**, supporting multiple modes or player categories within the same application.

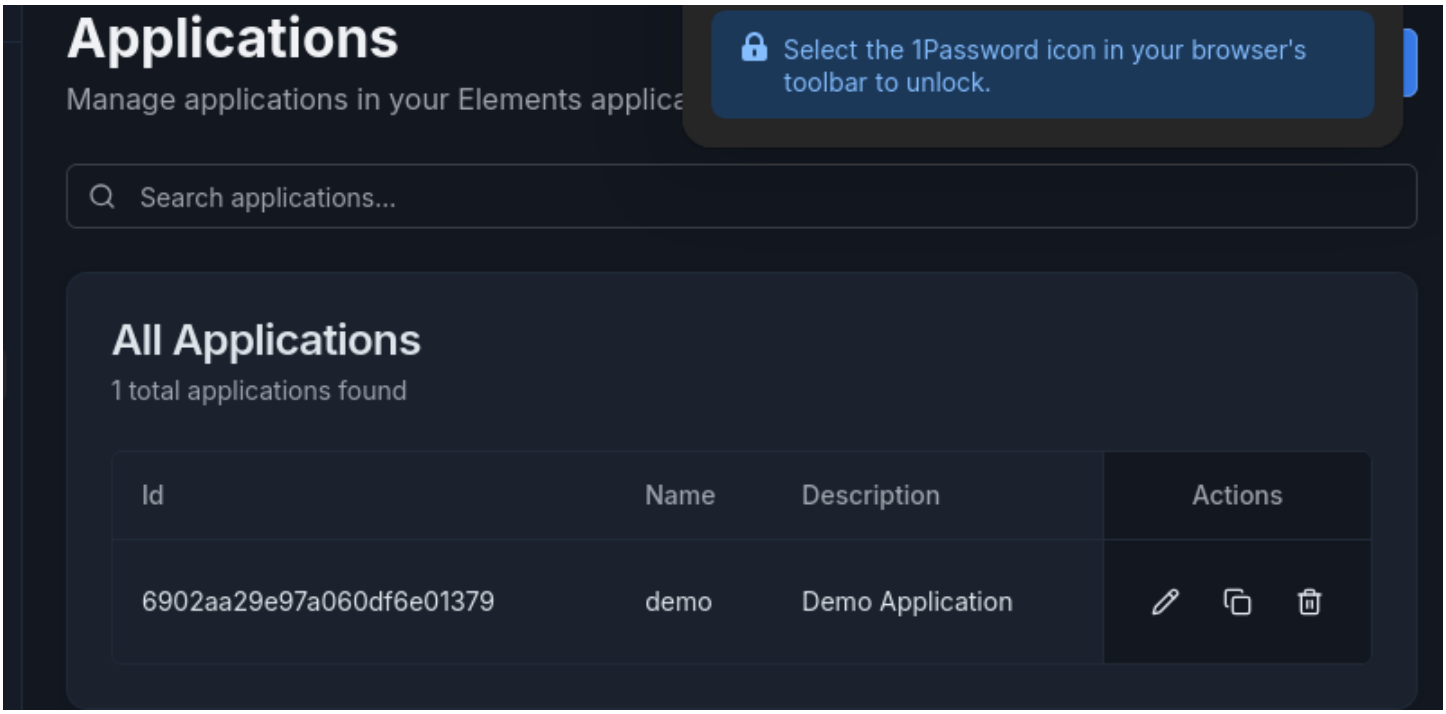
Updating a Matchmaking Configuration

Warning

Updating a matchmaking configuration in the UI or using the API does while a match is underway may invoke undefined behavior in the system resulting in lost games or inconsistent behavior. We recommend only modifying configurations during planned downtime or maintenance for games in production.

1. Open the Application for Editing

1. From the Namazu Elements dashboard, locate the application you want to modify (for example, *demo*).
2. In the **Actions** column, click the **edit (pencil)** icon to open the **Edit Application** dialog.



2. Edit the Matchmaking Configuration

1. In the **Edit Application** dialog, scroll down to the **Application Configurations** section. Each configuration shows its ID, name and type.
2. Locate the configuration of type **Matchmaking** (e.g., **unranked**) and click **Edit**. A new **Application Configuration** modal opens.
3. Modify the settings as needed:
   - **Name** – Provide a new unique name for the configuration (only letters, digits and underscores).
   - **Description** – Optionally describe the configuration.
   - **Max Profiles** – Enter the maximum number of player profiles allowed in a single match. This field defines the player count (minimum value is 2).
   - Additional settings such as **Linger Seconds**, **Timeout Seconds** and metadata options can be adjusted if required.
4. After making changes, click **Save Configuration**. A toast notification confirms “Configuration saved successfully,” and the modal closes.
5. Back in the **Edit Application** dialog, verify that the configuration list reflects the new name.
6. Click **Update Application** at the bottom-right of the dialog. This commits the changes to the application. A success message appears confirming the update.

Deleting a Configuration

Warning

Deleting a matchmaking configuration in the UI or using the API does not permanently delete the configuration. However, it does clear the name making it available for re-creation. This approach prevents existing matches from breaking due to database inconsistency. However, deleting a configuration while a match is underway may invoke undefined behavior in the system resulting in lost games. We recommend only deleting configurations during planned downtime or maintenance for games in production.

1. **Open the Application for Editing** and scroll to **Application Configurations**.
2. **Identify the Configuration** to delete. Confirm by reading its name and type.
3. **Click Remove**
    - In the **Actions** column for the configuration, click **Remove** (trash icon). The configuration disappears from the list, and a toast notifies that it was deleted successfully.
4. **Persist Changes**
    - Click **Update Application** to save the deletion. The **Edit Application** dialog closes and a success message confirms that the application was updated.

Summary

Matchmaking application configurations in Namazu Elements define how players are assembled into matches. Using the CMS, administrators can:

- **Create** new matchmaking configurations by setting a unique name, description, player count (Max Profiles), and optional timeout values.
- **Update** existing configurations to rename them, adjust player counts, or modify other parameters. Changes must be saved via **Save Configuration** followed by **Update Application**.
- **Delete** configurations when they are no longer needed, ensuring the application's configuration list stays current.

By following these steps, you can manage matchmaking behavior across different game modes within your application, leveraging Namazu's flexible MultiMatch API for scalable and customizable player experiences.