Access Lists

Overview

If you’re launching a new site or want to restrict parts of your site to a predefined list of users, then access lists or waitlists are probably a high priority for you.

Leveraging gating with access lists through the Dynamic dashboard is a simple no-code option that gives you flexibility to define various and manage various lists easily. Through your dashboard, you can:

1. Restrict site access based on a list of emails or wallet addresses.
2. Return a scope in the JWT of emails or wallet addresses.

In our dashboard, uou can design precise gating rules by combining multiple elements such as various access access list withNFT and token gates.

Usage

In theConfigurations page of your developer dashboard, click on the Access Control card.

Here you can create access lists based on emails or wallet addresses.

1. Click “Create new gate”
2. Name your gate
3. Select the gating method:1. Allow Site Access - this option will block users from access your site (we won’t generate a JWT) unless the criteria is met
4.
    1. Return scope - this option will not block users, but instead will return the JWT with a predefined scope if the user has met the defined criteria
5. Select whether you want to add an email or a wallet address.
6. Enter the email or wallet address.1. You can also add an alias to more easily keep track of these users.
7. Click the “Add +” button to save the user to the list. Keep adding users as needed.
8. Save and enable the toggle when you’re ready.
9. You’re done!

You can create multiple lists to help keep track of different groups of users (ie, VIPs, beta users, internal users, etc). Note: a user only needs to be in one of the lists to pass.

Working with scopes

To simplify working with scopes, we created a custom hook named useDynamicScopes. It allows you to easily checking the user scopes and to check for a specific scope on a user. To learn more about how to use this hook, see our docshere .

Customize the copy and button

You can customize the copy through props in our SDK by updating theaccessDeniedMessagePrimary andaccessDeniedMessageSecondary .

<DynamicContextProvider settings={{.... accessDeniedMessagePrimary: 'Your copy1', accessDeniedMessageSecondary: 'Your Copy2', }} You can also return a completely custom button by passing a JSX/TSX element to theaccessDeniedButton prop. Here’s an example to link to a contact page:

window.open(https://www.dynamic.xyz/talk-to-us,'_blank'), title: Book a demo } }}

The button should adhere to the following type:

type AccessDeniedCustomButton = { action?: () => void; title: string; };

Was this page helpful?