Using the Fee Grant Faucet

The Fee Grant faucet is backend appilication that allows you to give out fee grants to users.

This tutorial explores how to manage fee grants usingsecretjs . The Faucet serves as a backend service, granting, revoking, and querying fee allowances for blockchain addresses through a simple HTTP interface.

Follow these steps to install and run theFee Grant Faucet .

Prerequisites

- Git
- Node.js
- npm (Node Package Manager)
-

Clone the Repository

Clone the repository to your local machine and navigate to the project directory:

```
Copy bash gitclonehttps://github.com/SecretSaturn/FeeGrantFaucet2.0 cdFeeGrantFaucet2.0
```

Install Dependencies

Install the necessary Node.js dependencies:

```
Copy npminstall
```

Configure Environment Variables

Create a .env file in the root of the project directory. Use the .env.sample provided in the repository as a reference and fill in the necessary details. Ensure these variables are correctly set in your.env file for the script to function.

Before diving into the code, it's essential to set up the environment:

- CHAIN_ID
- : Identifier of the blockchain network.
- LCD_NODE
- : URL of the Light Client Daemon (LCD) for interacting with the blockchain.
- FAUCET_MNEMOMIC
- : Mnemonic for the wallet used in transactions.
-

Don't forget to fill up your Faucet with SCRT.

Run the Application

Start the application:

```
Copy npmrunstart
```

Express Server Initialization

The script uses Node Express, a popular Node.js framework, to create a web server:

- Endpoint Creation
- : An endpoint/claim/:address
- is set up. It's designed to handle fee grants for a specified blockchain address passed as a parameter.
-

Fee Grant Management

The script includes robust functionality for managing fee grants:

- Address Validation
- : UsesvalidateAddress
- to ensure the provided blockchain address is valid.
- Querying Balance and Fee Grants
- : Before granting a fee, the script can check the existing balance and any active fee grants for the address.
-

Granting and Revoking Allowances

The core functionality of the script lies in managing fee grants:

- Function
- giveFeeGrant
- : Depending on the situation, this function either revokes an old fee grant and issues a new one or issues a new grant directly.
- Message Construction
- :
-
  - MsgRevokeAllowance
-
  - : Used to revoke an existing fee grant.
-
  - MsgGrantAllowance
-
  - : Used to grant a new fee allowance to an address.
- *
-

Transaction Broadcasting and Response

Handling transactions is a critical part of the script:

- Broadcasting Transactions
- : Utilizessecretjs.tx.broadcast
- to send the transaction to the blockchain.
- Transaction Parameters
- : Includes memo, fee, and gas limit details.
- JSON Output
- : The script outputs a JSON object with thefeegrant
- details, including grantee, granter, spend limit, and expiration.
-

Error Handling

The script incorporates error handling mechanisms:

- Invalid Addresses
- : Returns an error if the provided address is invalid.
- Transaction Failures
- : Catches and logs transaction-related errors.
-

Code Snippets and Examples

Here are some key code snippets and example responses:

- Endpoint
- /claim/:address
- :

-
  ```
- Copy
- app.get("/claim/:address",async(req,res)=>{
- // ... (faucet logic)
- returnres.json(results);
- });
- ```
  ```

- Example JSON Response from the faucet
- :

-
  ```
- Copy
- {
- "feegrant":{
- "grantee":"secret1...",
- "granter":"secret1...",
- "spend_limit":[{"amount":"10000","denom":"uscrt"}],
- "expiration":"2021-12-31T23:59:59"
- },
- "address":"secret1..."
- }
- ```
-

Last updated2 months ago On this page *Prerequisites * Clone the Repository * Install Dependencies * Configure Environment Variables * Run the Application * Express Server Initialization * Fee Grant Management * Granting and Revoking Allowances * Transaction Broadcasting and Response * Error Handling * Code Snippets and Examples

Was this helpful? Edit on GitHub Export as PDF
  ```