OpenSea API (Schema + Resolver + Server)

Group members:Xingen Chen (xc2641@columbia.edu), Zhengxuan Wen (zw2851@columbia.edu), Fangzheng Wu (fw2396@columbia.edu), Bo Yu (by2345@columbia.edu)

1. Introduction

In recent years, the world of cryptocurrencies has witnessed a significant paradigm shift, with the advent of non-fungible tokens (NFTs). NFTs represent unique digital assets, stored on the blockchain, that have gained immense popularity due to their application in various sectors such as art, gaming, and collectibles. OpenSea, as one of the largest NFT marketplaces built on the Ethereum blockchain, has become a focal point for the trade and discovery of these unique assets.
Our implementation outlined in this document demonstrates how to interact with the OpenSea API, a powerful tool designed to retrieve and engage with NFT information on the Ethereum blockchain. This API allows developers and users to access a variety of data related to OpenSea NFTs, including basic information, creator details, transaction history, metadata, owner balance, and top token holders.
By harnessing the power of the OpenSea API, users can unlock new potential in the burgeoning world of NFTs, bringing increased transparency, efficiency, and innovation to the digital asset space.

2. Implementation

2.1 System Overview

This version of implementation for the OpenSea API uses the structure instructed in the original pdf file. The API is designed to retrieve and interact with OpenSea NFT information on the Ethereum blockchain. Server.js sets up a simple GraphQL server using the Apollo Server library. This server will enable clients to fetch and manipulate data by sending GraphQL queries and mutations to the server's API endpoint. The schema is responsible for defining the data types and relationships that allow users to interact with and retrieve information about NFTs on the platform.
The primary use cases of the API in the context of OpenSea NFTs are:

- Retrieve OpenSea NFT basic information: The API allows users to fetch essential data such as token ID, token URI, owner address, contract address, name, symbol, and approved address.
- Fetch OpenSea NFT creator information: Users can retrieve the creator address of an NFT listed on OpenSea, which is the original address that minted the token.
- Obtain OpenSea NFT transaction history: The API provides the ability to fetch past transfer transactions for a specific NFT listed on OpenSea, including sender and receiver addresses and transaction hashes.
- Access OpenSea NFT metadata: Users can retrieve metadata associated with an OpenSea NFT, such as the name, description, and image URL.
- Query OpenSea NFT owner balance: The API allows users to fetch the balance of a particular NFT owner on OpenSea, which represents the number of tokens owned by the address.
- Discover top OpenSea NFT token holders: Users can obtain a list of top token holders for a specific contract listed on OpenSea, ranked by the number of tokens they own. The API allows users to specify the number of top holders to retrieve.

2.2 Schema.graphql

In the proposed schema, a GraphQL API is designed to model and query digital assets data. The schema comprises a root query type and several custom data types to represent the various aspects of the digital assets.

- Query: The root query type contains a single query named ‘assetById’, which accepts two mandatory arguments, ‘contractAddress’ and ‘tokenId’, both of type String. The query returns an object of the custom type Asset.
- Asset: This custom type signifies an asset and encompasses various fields relevant to the asset, such as ‘tokenId’, ‘tokenURI’, ‘owner’, ‘contractAddress’, ‘name’, ‘symbol’, ‘approvedAddress’, ‘creator’, ‘transactions’, ‘metadata’, ‘ownerBalance’, and ‘topTokenHolders’. Most of these fields have scalar types (String or Int), while ‘transactions’ and ‘topTokenHolders’ are lists containing custom types (Transaction and TokenHolder, respectively).
- Transaction: Representing a transaction event, this custom type contains details such as the transaction ‘id’, the sender (‘from’), the receiver (‘to’), and the ‘transactionHash’.
- Metadata: This custom type represents the metadata associated with a particular asset. It includes fields such as ‘name’, ‘description’, and ‘image’.
- TokenHolder: A custom type characterizing a token holder, it contains the holder's Ethereum address and the number of tokens (‘tokenCount’) they possess.

2.3 Resolvers.js

The resolver function we have is in JavaScript format which uses Web3.js as a library, which is used to interact with Ethereum blockchain for ERC 721 tokens. It handles GraphQL queries and returns the requested data from the blockchain. The resolver functions are organized into our main Query : Asset. Within the resolver functions, the assetById function is responsible for fetching information about a specific ERC721 token using its contractAddress and tokenId. This is done by first initializing a new instance of the web3.eth.Contract class, utilizing the ERC721_ABI (Application Binary Interface) which is stored in a separate JSON file. Then, various properties of the token are retrieved through the use of methods provided by the ERC721 contract, such as ownerOf, tokenURI, name, symbol, and getApproved. After gathering this information, the function creates and returns an object that contains the relevant token properties.

Several functions within the Asset resolver are utilized to resolve properties for an Asset object within the GraphQL schema. This particular object represents an ERC721 token and includes a variety of properties, such as owner, creator, tokenId, tokenURI, metadata, ownerBalance, and topTokenHolders. To retrieve this information, each resolver function accesses the blockchain via the web3.eth.Contract class and the provided ERC721 contract methods.

The findCreator function is a function used to find the creator of the NFT. It retrieves the address of the creator of an ERC721 token by looking for the first Transfer event from the zero address (0x0000000000000000000000000000000000000000) to another address.

The fetchTransferEvents function is a function used to find transaction history of the target NFT. It retrieves the Transfer events for a given ERC721 token using the getPastEvents method provided by the ERC721 contract.

The fetchMetadata function is a function used to find metadata of target NFT. It retrieves the metadata for a given ERC721 token by fetching the JSON data from the tokenURI. It able to tell us the description of the NFT as well as a quick image jpg link to the NFT (Example will be shows on the test result below)

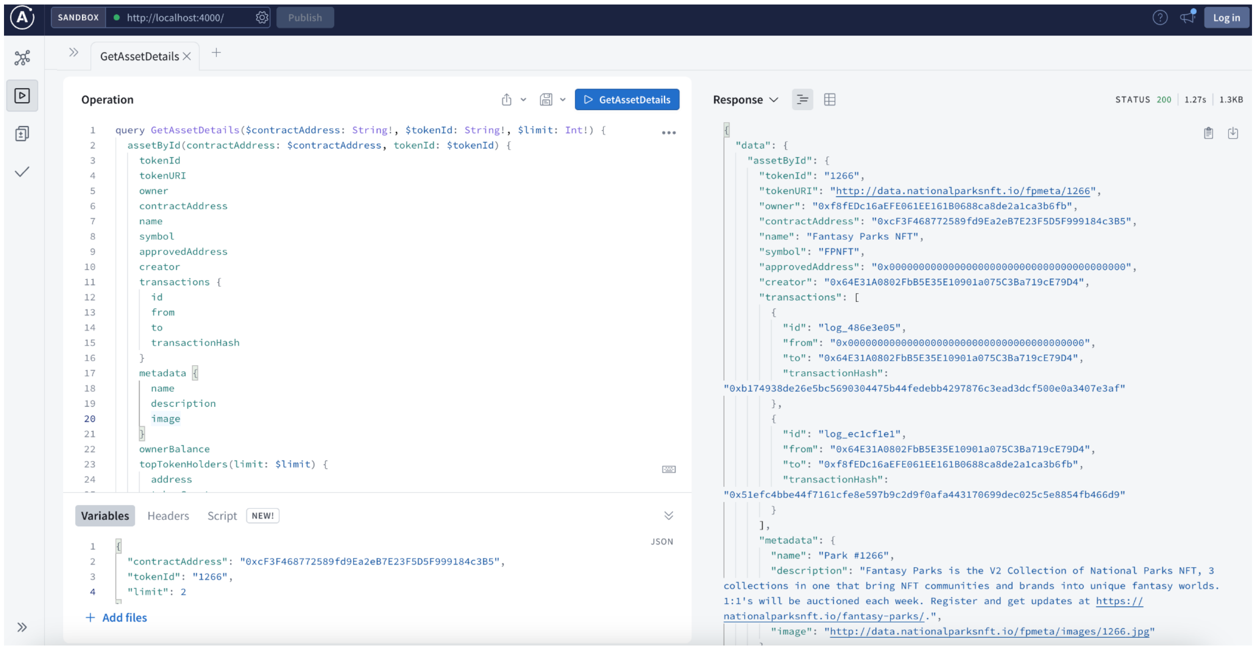
The fetchTopTokenHolders function is a function retrieves the top N token holders for a given ERC721 contract by processing all the Transfer events and counting the number of tokens held by each address where we going to control the N by the input “limit”

2.3 Server.js

The server uses the Apollo Server library and Node.js. The server is configured with a GraphQL schema and resolver functions. Required modules, such as ‘apollo-server’ and ‘fs’, are imported, along with the resolvers module. The ‘schema.graphql’ file containing the GraphQL schema is read synchronously and parsed into a valid GraphQL schema using the ‘gql’ template literal function. A new Apollo Server instance is created with the parsed schema and resolver functions as configuration options. Finally, the server starts using the ‘server.listen()’ function, which listens for incoming requests and logs the server's URL once it is ready. This server enables clients to query and manipulate data based on the defined schema and resolver functions.

3. Test Results

Below is a test of using the API to query information of the following NFT:
https://opensea.io/assets/ethereum/0xcf3f468772589fd9ea2eb7e23f5d5f999184c3b5/1266

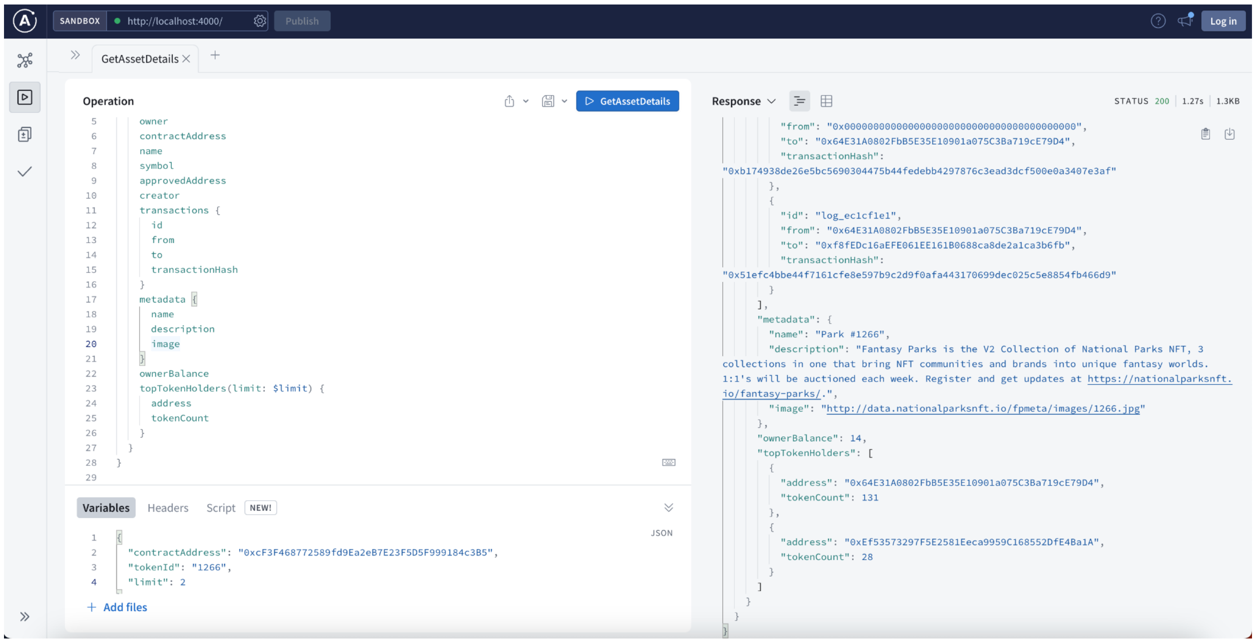


```
query GetAssetDetails($contractAddress: String!, $tokenId: String!, $limit: Int!) {
  assetById(contractAddress: $contractAddress, tokenId: $tokenId) {
    tokenId
    tokenURI
    owner
    contractAddress
    name
    symbol
    approvedAddress
    creator
    transactions {
      id
      from
      to
      transactionHash
    }
    metadata {
      name
      description
      image
    }
    ownerBalance
    topTokenHolders(limit: $limit) {
      address
```

```
{
  "contractAddress": "0xcF3F468772589fd9Ea2eB7E23F5D5F999184c3B5",
  "tokenId": "1266",
  "limit": 2
```

```
{
  "data": {
    "assetById": {
      "tokenId": "1266",
      "tokenURI": "http://data.nationalparksnft.io/fpmeta/1266",
      "owner": "0xf8fEDc16aEFE061EE161B0688ca8de2a1ca3b6fb",
      "contractAddress": "0xcF3F468772589fd9Ea2eB7E23F5D5F999184c3B5",
      "name": "Fantasy Parks NFT",
      "symbol": "FPNFT",
      "approvedAddress": "0x0000000000000000000000000000000000000000",
      "creator": "0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4",
      "transactions": [
        {
          "id": "log_486e3e05",
          "from": "0x0000000000000000000000000000000000000000",
          "to": "0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4",
          "transactionHash":
"0xb174938de26e5bc5690304475b44fedebb4297876c3ead3dcf500e0a3407e3af"
        },
        {
          "id": "log_ec1cf1e1",
          "from": "0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4",
          "to": "0xf8fEDc16aEFE061EE161B0688ca8de2a1ca3b6fb",
          "transactionHash":
"0x51efc4bbe44f7161cfe8e597b9c2d9f0afa443170699dec025c5e8854fb466d9"
        }
      ],
      "metadata": {
        "name": "Park #1266",
        "description": "Fantasy Parks is the V2 Collection of National Parks NFT, 3
collections in one that bring NFT communities and brands into unique fantasy worlds.
1:1's will be auctioned each week. Register and get updates at https://
nationalparksnft.io/fantasy-parks/.",
        "image": "http://data.nationalparksnft.io/fpmeta/images/1266.jpg"
```

Figure 1: query test part 1

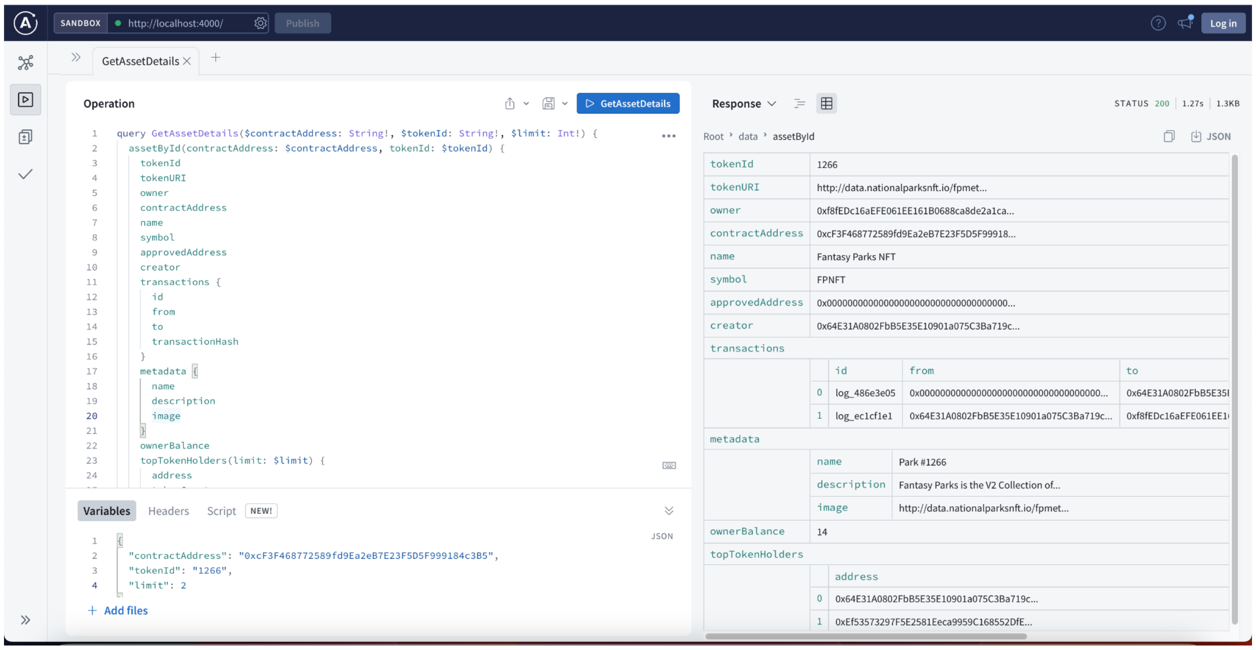


```graphql
    owner
    contractAddress
    name
    symbol
    approvedAddress
    creator
    transactions {
      id
      from
      to
      transactionHash
    }
    metadata {
      name
      description
      image
    }
    ownerBalance
    topTokenHolders(limit: $limit) {
      address
      tokenCount
    }
  }
}
```

```json
{
  "contractAddress": "0xcF3F468772589fd9Ea2eB7E23F5D5F999184c3B5",
  "tokenId": "1266",
  "limit": 2
}
```

```json
          "from": "0x0000000000000000000000000000000000000000",
          "to": "0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4",
          "transactionHash":
"0xb174938de26e5bc5690304475b44fedebb4297876c3ead3dcf500e0a3407e3af"
        },
        {
          "id": "log_ec1cf1e1",
          "from": "0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4",
          "to": "0xf8fEDc16aEFE061EE161B0688ca8de2a1ca3b6fb",
          "transactionHash":
"0x51efc4bbe44f7161cfe8e597b9c2d9f0afa443170699dec025c5e8854fb466d9"
        }
      ],
      "metadata": {
        "name": "Park #1266",
        "description": "Fantasy Parks is the V2 Collection of National Parks NFT, 3
collections in one that bring NFT communities and brands into unique fantasy worlds.
1:1's will be auctioned each week. Register and get updates at https://nationalparksnft.
io/fantasy-parks/.",
        "image": "http://data.nationalparksnft.io/fpmeta/images/1266.jpg"
      },
      "ownerBalance": 14,
      "topTokenHolders": [
        {
          "address": "0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4",
          "tokenCount": 131
        },
        {
          "address": "0xEf53573297F5E2581Eeca9959C168552DfE4Ba1A",
          "tokenCount": 28
        }
      ]
    }
  }
}
```

Figure 2: query test part 2



```graphql
query GetAssetDetails($contractAddress: String!, $tokenId: String!, $limit: Int!) {
  assetById(contractAddress: $contractAddress, tokenId: $tokenId) {
    tokenId
    tokenURI
    owner
    contractAddress
    name
    symbol
    approvedAddress
    creator
    transactions {
      id
      from
      to
      transactionHash
    }
    metadata {
      name
      description
      image
    }
    ownerBalance
    topTokenHolders(limit: $limit) {
      address
```

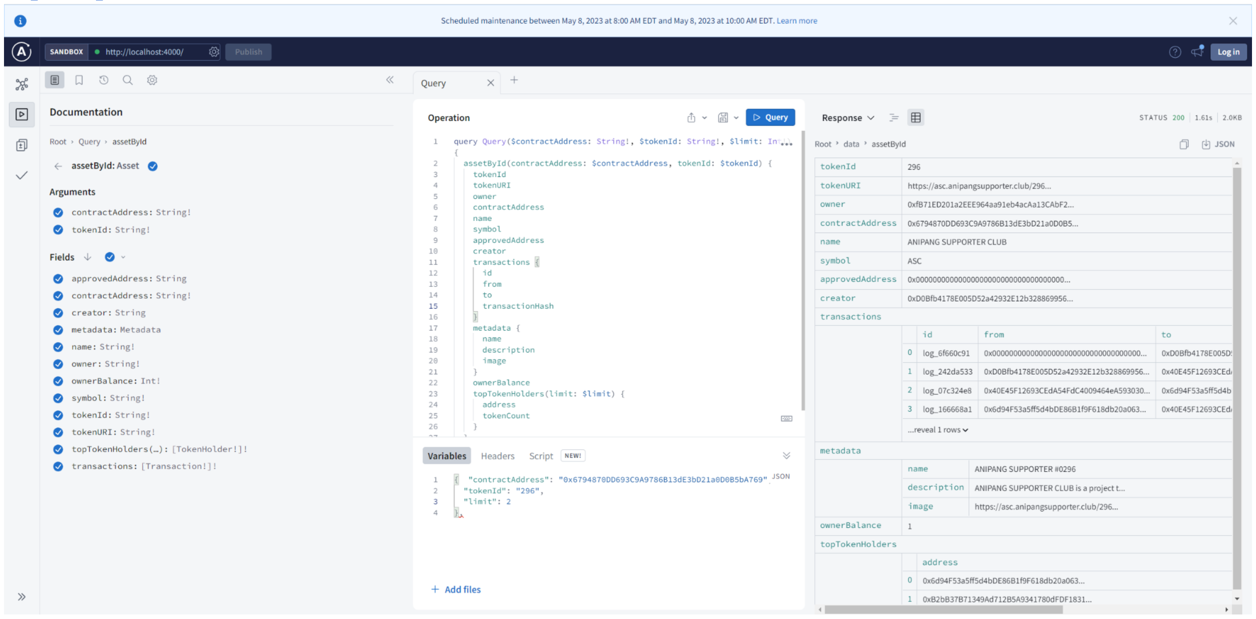
```json
{
  "contractAddress": "0xcF3F468772589fd9Ea2eB7E23F5D5F999184c3B5",
  "tokenId": "1266",
  "limit": 2
}
```

tokenId	1266			
tokenURI	http://data.nationalparksnft.io/fpmet...			
owner	0xf8fEDc16aEFE061EE161B0688ca8de2a1ca...			
contractAddress	0xcF3F468772589fd9Ea2eB7E23F5D5F99918...			
name	Fantasy Parks NFT			
symbol	FPNFT			
approvedAddress	0x000000000000000000000000000000000...			
creator	0x64E31A0802FbB5E35E10901a075C3Ba719c...			
transactions				
		id	from	to
	0	log_486e3e05	0x000000000000000000000000000000000...	0x64E31A0802FbB5E35I
	1	log_ec1cf1e1	0x64E31A0802FbB5E35E10901a075C3Ba719c...	0xf8fEDc16aEFE061EE1
metadata				
	name	Park #1266		
	description	Fantasy Parks is the V2 Collection of...		
	image	http://data.nationalparksnft.io/fpmet...		
ownerBalance	14			
topTokenHolders				
		address		
	0	0x64E31A0802FbB5E35E10901a075C3Ba719c...		
	1	0xEf53573297F5E2581Eeca9959C168552DfE...		

Figure 3: query test part 3

The test results successfully demonstrate the use of the API to query information about an NFT listed on OpenSea. The NFT is found at the following URL: https://opensea.io/assets/ethereum/0xcf3f468772589fd9ea2eb7e23f5d5f999184c3b5/1266. The API returns accurate information about the NFT, such as the tokenId (1266), tokenURI (http://data.nationalparksnft.io/fpmeta/1266), and owner address (0xf8fEDc16aEFE061EE161B0688ca8de2a1ca3b6fb). Additionally, the API provides contract-related

details, including the contractAddress (0xcF3F468772589fd9Ea2eB7E23F5D5F999184c3B5), name (Fantasy Parks NFT), and symbol (FPNFT).
The creator of the NFT is confirmed to be the address 0x64E31A0802FbB5E35E10901a075C3Ba719cE79D4. The API also fetches the NFT's transaction history, revealing two transactions: one for the NFT creation and the other for transferring the NFT to its current owner.
The metadata part represents an NFT titled "Park #1266" from the "Fantasy Parks" collection, which is the Version 2 (V2) collection of the National Parks NFT project. The project combines three distinct collections into a single, unique fantasy world, creating a bridge between NFT communities and brands. Furthermore, the API returns the balance of the NFT's owner, confirming they own 14 tokens. Lastly, the API lists the top token holders for the contract, with the creator holding the most tokens (131) and another address (0xEf53573297F5E2581Eeca9959C168552DfE4Ba1A) holding 28 tokens. Overall, the API effectively retrieves valuable information about the OpenSea NFT, making it an essential tool for accessing and analyzing data related to NFTs listed on the platform.
Another successful test has been conducted utilizing the API to examine a different NFT listed on OpenSea. The NFT can be accessed at the following URL: https://opensea.io/assets/ethereum/0x6794870dd693c9a9786b13de3bd21a0d0b5ba769/296.



assetById: Asset

Arguments

- contractAddress: String!
- tokenId: String!

Fields

- approvedAddress: String
- contractAddress: String!
- creator: String
- metadata: Metadata
- name: String!
- owner: String!
- ownerBalance: Int!
- symbol: String!
- tokenId: String!
- tokenURI: String!
- topTokenHolders(…): [TokenHolder!]!
- transactions: [Transaction!]!

```
query Query($contractAddress: String!, $tokenId: String!, $limit: In
{
  assetById(contractAddress: $contractAddress, tokenId: $tokenId) {
    tokenId
    tokenURI
    owner
    contractAddress
    name
    symbol
    approvedAddress
    creator
    transactions {
      id
      from
      to
      transactionHash
    }
    metadata {
      name
      description
      image
    }
    ownerBalance
    topTokenHolders(limit: $limit) {
      address
      tokenCount
    }
```

```
{ "contractAddress": "0x6794870DD693C9A9786B13dE3bD21a0D0B5bA769",
  "tokenId": "296",
  "limit": 2
}
```

tokenId	296
tokenURI	https://asc.anipangsupporter.club/296...
owner	0xfB71ED201a2EEE964aa91eb4acAa13CAbF2...
contractAddress	0x6794870DD693C9A9786B13dE3bD21a0D0B5...
name	ANIPANG SUPPORTER CLUB
symbol	ASC
approvedAddress	0x00000000000000000000000000000000...
creator	0xD0Bfb4178E005D52a42932E12b328869956...

transactions

	id	from	to
0	log_6f660c91	0x00000000000000000000000000000000...	0xD0Bfb4178E005D
1	log_242da533	0xD0Bfb4178E005D52a42932E12b328869956...	0x40E45F12693CEd
2	log_07c324e8	0x40E45F12693CEdA54FdC4009464eA593030...	0x6d94F53a5ff5d4b
3	log_166668a1	0x6d94F53a5ff5d4bDE86B1f9F618db20a063...	0x40E45F12693CEd

...reveal 1 rows

metadata

name	ANIPANG SUPPORTER #0296
description	ANIPANG SUPPORTER CLUB is a project t...
image	https://asc.anipangsupporter.club/296...

ownerBalance 1

topTokenHolders

	address
0	0x6d94F53a5ff5d4bDE86B1f9F618db20a063...
1	0xB2bB37B71349Ad712B5A9341780dFDF1831...

Figure 4: query test a different NFT

4. Project Files

Github repo: https://github.com/zhengxuanwen/OpenSea_API_ELEN6883