Overview

The parameters and return values of this API series are in JSON format unless otherwise specified.

Authentication

The authentication for all Staking APIs uses the Keccak256 signature method. Users need to upload the address used for signing to the Staking platform, and then pass the following content in the header.

```
Authorization: Quark-Keccak256-ECDSA
Address=0x1234567890abcdef1234567890abcdef12345678,
Nonce=1cb89e48-5d71-43e6-ba61-4a810b879e5e,
Timestamp=1686048892,
Signature=<ECDSA Sign for Keccak256(Nonce(string) + Timestamp(int256))>
```

Parameter Explanation

- Address: Wallet address. An error will occur if it doesn't exist in the system.
- Nonce: Random string. It's recommended to use UUID. The same user's Nonce cannot be used multiple times within 24 hours. If it's repeated, an error will occur.
- Timestamp: Unix timestamp of the current time in seconds. If this time is earlier than the server time by 10 minutes, an error will occur.
- Signature: Hash the Nonce and Timestamp using Keccak256, then perform an ECDSA signature. If the signature cannot be matched with an Address or cannot be resolved, an error will occur.

Deposit Interface

Deposit Assign

URL: `/deposits/{txHash}/assignment`

Method: `POST`

Authentication Required: YES

Path Parameters:

- txHash: Deposit Transaction Hash

Parameters:

- assignments: Assignment list

Note that the sum of validatorCount cannot exceed the total deposit amount of this transaction, otherwise an error will occur.

Parameters Example

```
{
        "assignments": [{
                "uid": "1",
                "validatorCount": 5
        },{
                "uid": "2",
                "validatorCount": 5
        }]
}
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.

- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: UserValidators list

Content example:

```
{
    "code": 0,
    "msg": "ok",
    "data": [
            {
                    "uid": "1",
                    "validators": [{
                                "validatorIndex": 1234567,
                                "pubkey": "0x1234567890"
                    },{
                                "validatorIndex": 1234568,
                                "pubkey": "0x1234567891"
                    }]
            }
    ]
}
```

Batch Deposit Assign

URL: `/deposits_assignments`

Method: `POST`

Authentication Required: YES

Parameters:

- assignments: Assignment list
- txHashs: txHash list

Note that the sum of validatorCount must not exceed the total deposit amount of this transaction, otherwise an error will occur.

Parameters Example

```
{
        "assignments": [{
                "uid": "1",
                "validatorCount": 5
        },{
                "validatorCount": 5
        }],
        "tx_hashs": ["0x3f8d4a00bf20a0025d109ff4fe8034e429f05d0f1a579b94b7d806a3d259f0e9","0xfa84ea1ffde16ee2a930798d4311557e60338e3e28e30b288fac1bd4fab656a2"]
}
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: UserValidators list

Content example:

```
{
    "code": 0,
    "msg": "ok",
    "data": [
            {
                    "uid": "1",
                    "validators": [{
                                "validatorIndex": 1234567,
                                "pubkey": "0x1234567890"
                    },{
                                "validatorIndex": 1234568,
                                "pubkey": "0x1234567891"
                    }]
            }
    ]
}
```

Deposit List

URL: `/deposits`

Method: `GET`

Authentication Required: YES

Query Parameters:

- pageNum: Page number, starting from 1.
- pageSize: Number of items displayed per page.

Query Parameters Example

```
pageNumber=1&pageSize=10
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data:
  - pageNum: Page number, starting from 1.
  - pageSize: Number of items displayed per page.
  - total: Data volume
  - list: Deposit list

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "pageNum": 1,
                "pageSize": 1,
                "total": 100,
                "list": [{
                        "id": 1,
                        "brokerId": 1,
                        "fromAddress": "0x1234567890",
                        "depositAddress": "0x1234567890",
                        "contractAddress": "0x1234567890",
                        "txHash": "0x1234567890",
                        "chainName": "ethereum",
                        "blockHeight": 12345678,
                        "slot": 2345678,
                        "blockTime": 1686070859,
                        "type":1,
                }]
        }
}
```

Deposit Detail

URL: `/deposits/{txHash}`

Method: `GET`

Authentication Required: YES

Query Parameters:

- pageNum: Page number, starting from 1.
- pageSize: Number of items displayed per page.

Query Parameters Example

```
pageNumber=1&pageSize=10
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: Deposit

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "id": 1,
                "brokerId": 1,
                "fromAddress": "0x1234567890",
                "depositAddress": "0x1234567890",
                "contractAddress": "0x1234567890",
                "txHash": "0x1234567890",
                "chainName": "ethereum",
                "blockHeight": 12345678,
                "slot": 2345678,
                "blockTime": 1686070859,
                "validators": [{
                        "uid": "1",
                        "validatorIndex": 1234567,
                        "pubkey": "0x1234567890",
                        "balance": "32030000000",
                        "status": "active",
                        "activationTime": "2023/05/04 15:04:05",
                        "exitTime": "",
                        "principal": "32000000000"
                }]
        }
}
```

Get Deposit Data

URL: `/deposits/deposit_data`

Method: `POST`

Authentication Required: YES

Parameters:

- uid: User's UID
- quantity: Total number of nodes
- withdrawalAddress: Withdrawal address of the nodes

Parameters Example

```
{

        "uid": "1",
    "quantity": 5,
    "withdrawalAddress": "0x3435tre342346543234"
}
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: DepositData

Content example:

```
{
        "code": 0,
        "msg": "ok",
    "data": {
    "ethereum": {
      "contractAddress": "0xeC98Ac89Df4F62FA6fc79ef67309E1b436342d34",
      "depositData": [
        {
          "depositDataRoot": "",
          "pubkey": "0x234353t4retrew423445334",
          "signature": "0xrgdfgfvcfgr3ewrfrfdfd",
          "withdrawalCredential": "0x2343ewr2343efdfer33er"
        }
      ],
      "estimatedGas": 0,
      "unsignedTransaction": ""
    },
    "network": "mainnet",
    "protocol": "",
    "period":"2023-09-14T10:32:30.762098195Z"  // expire time
  },
}
```

Sending Transaction

URL: `/deposits

Method: `POST`

是否需要鉴权: YES

Parameters:

- signedTransaction: Signed transaction data
- chainName: ethereum

```
// sign progress:
contractAddress :=
common.HexToAddress("0x3C98Ac89Df4F62FA6fc79ef67309E1b436342dc7")
fromAddress := common.HexToAddress("0x2B3778B253dB55B98eCED3EF427992740C17db17")
nonce, err := client.PendingNonceAt(ctx, fromAddress)
if err != nil {
    return
}

unsigned :=
"0x6225d68c000000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000
000000000000000000000000000000c000000000000000000000000000000000000000000000000000000000000001600000000000
00000000000000000000000000000000000000000000000001e0000000000000000000000000000000000000000000000000000000
00000002a07400000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000100000000000000000000000000000000000000000000000000000000000000020000000000000
0000000000000000000000000000000000000000000000000030aeb2d5ba5c2c84f169f0e07c318ea52143909b6916decfd6e06ff23
0cb4d719d08dd8328986ed77773506d920940b2a4000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000100000000000000000000000000000000000000000000000000000000000000200000000000000
0000000000000000000000000000000000000000000000000200100000000000000000000002b3779a253db55b98eced3ef42799274
0c17db1700000000000000000000000000000000000000000000000000000000000000010000000000000000000000000000000000
00000000000000000000000000020000000000000000000000000000000000000000000000000000000000000000060a60e5e19ef7b34
ac1a95b0bc2424f2b48d50b71e2c058445e3af04cbe2a82998800856c56f03ed87db389d22053a0b0411239b653f979cc79dc7bf8d0
9eb229e0e6b52d1f421bfa7c8036c3dc3fa14bbdd66edeed08c72c88b13c5bbf8f0df15000000000000000000000000000000000000
00000000000000000000000000011a229df924fa224755c55ca222220564d4c6194c5aa315dea2cb98750e63b90a"
estimateGas, err := client.EstimateGas(ctx, ethereum.CallMsg{
    From: fromAddress,
    To: &contractAddress,
    Gas: 0,
    GasPrice: nil,
    GasFeeCap: big.NewInt(5),
    GasTipCap: big.NewInt(10),
    Value: eth_convert.EtherToWei(big.NewInt(32)),
    Data: common.FromHex(unsigned),
})
if err != nil {
    return
}

tx := types.NewTx(&types.DynamicFeeTx{
    Nonce: nonce,
    GasTipCap: big.NewInt(5),
    GasFeeCap: big.NewInt(10),
    Gas: estimateGas,
    To: &contractAddress,
    Value: eth_convert.EtherToWei(big.NewInt(32)),
    Data: common.FromHex(unsigned),
})

privateKeyHex := "privateKey"
// decrypt private key
privateKey, err := crypto.HexToECDSA(privateKeyHex)
if err != nil {
    return
}

// sign by private key
chainID := big.NewInt(32382)
signedTx, err := types.SignTx(tx, types.LatestSignerForChainID(chainID), privateKey)
if err != nil {
    return
}

txBytes, err := rlp.EncodeToBytes(signedTx)
signedTxHex := fmt.Sprintf("0x%x", txBytes)
```

Parameters Example

```
{

        "signedTransaction":
"0xb9035802f90354827e7e08050a8301e67b94ec98ac89df4f62fa6fc79ef67309e1b436342dc78901bc16d674ec800000b902e462
25d68c000000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000000000
0000000000000000000000000c00000000000000000000000000000000000000000000000000000000000000160000000000000000000
0000000000000000000000000000000000000000000001e000000000000000000000000000000000000000000000000000000000000
002a074000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000010000000000000000000000000000000000000000000000000000000000000002000000000000000000
000000000000000000000000000000000000000000000030aeb2d5ba5c2c84f169f0e07c318ea52143909b6916decfd6e06ff230cb4d
719d08dd8328986ed77773506d920940b2a400000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000100000000000000000000000000000000000000000000000000000000000000200000000000000000000
00000000000000000000000000000000000000000000200100000000000000000000002b3779a253db55b98eced3ef427992740c17d
b17000000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000000000000
000000000000000000000020000000000000000000000000000000000000000000000000000000000000000060a60e5e19ef7b34ac1a9
5b0bc2424f2b48d50b71e2c058445e3af04cbe2a82998800856c56f03ed87db389d22053a0b0411239b653f979cc79dc7bf8d09eb22
9e0e6b52d1f421bfa7c8036c3dc3fa14bbdd66edeed08c72c88b13c5bbf8f0df1500000000000000000000000000000000000000000
000000000000000000000011a229df924fa224755c55ca222220564d4c6194c5aa315dea2cb98750e63b90ac001a022574976088742
2552a2c3c42be17a885e2c64715ff7fba119b89f7cd569f4e8a01844886ca32c0ea0a454659abeeec5d8193c45cd6ab0943d13c5c7f
ab15c10a0",
    "chainName": "ethereum",
}
```

Success Response

Code: `200 OK`

Response Definition:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data:
  - txHash: Transaction Hash

Content example:

```
 {
        "code": 0,
        "msg": "ok",
        "data": {
                "txHash": "0xae3b8e113b0537939cffcb0862e96d6aad86df775178871bd825f63db46c8fd4"
        }
}
```

User-related APIs

Create New User

URL: `/users`

Method: `POST`

Authentication Required: YES

Parameters:

User

Parameters Example

```
{
    "uid": "1",
    "email": "rothcold1@example.com",
    "address": "0x123456",
    "depositAddresses": ["0x1234","0x4321"]
}
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: User

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "uid": "User-1",
                "email": "example@example.com",
                "address": "0x1234567890"
        }
}
```

Query User List

URL: `/users`

Method: `GET`

Authentication Required: YES

Query Parameters:

- pageNum: Page number, starting from 1.
- pageSize: Number of items displayed per page.

Query Parameters Example

```
pageNumber=1&pageSize=10
```

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data:
  - pageNum: Page number, starting from 1.
  - pageSize: Number of items displayed per page.
  - total: Data volume
  - list: Deposit list

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "pageNum": 1,
                "pageSize": 1,
                "total": 100,
                "list":[{
                        "uid": "User-1",
                        "email": "example@example.com",
                        "address": "0x1234567890"
                }]
        }
}
```

Query User Details

URL: `/users/{uid}`

Method: `GET`

Authentication Required: YES

Path Parameters:

- uid: user ID

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: UserDetails

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "uid": "User-1",
                "email": "example@example.com",
                "address": "0x1234567890",
                "validators": [{
                        "uid": "1",
                        "validatorIndex": 1234567,
                        "pubkey": "0x1234567890",
                        "balance": "32030000000",
                        "status": "active",
                        "activationTime": "2023/05/04 15:04:05",
                        "exitTime": "",
                        "principal": "32000000000"
                }]
        }
}
```

Delete User (User must have no nodes or all nodes must be in exited state)

URL: `/users/{uid}`

Method: `DELETE`

Authentication Required: YES

Path Parameters:

- uid: user ID

Success Response

Code: `200 OK`

Definition of Return Values:

- code: number, 0 indicates success; otherwise, it indicates an error.
- msg: If successful, the response will be 'ok'; otherwise, it will be an error message.
- data: If successful, it will return "success"; otherwise, it will be null.

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": "success"
}
```

Node-related APIs

Get Node Information

Used to retrieve node information.

URL: `/validators`

Method: `GET`

Auth required: YES

Query Parameters:

- pageSize: Page size
- pageNum: Page number, starting from 1
- uid: User ID to which the node belongs; if not provided, it will query all node information under the Broker.

Query Parameters Example

```
pageSize=10&pageNum=1
```

Success Response

Code: `200 OK`

Content Definition:

- code: number, 0 is success, others are error
- msg: if response success, it will be "ok", otherwise, it will be error message
- data:
  - pageSize: Page size
  - pageNum: Page number
  - total: Data volume
  - list: ValidatorWithDetails list

Content example:

```
{
    "code": 0,
    "msg": "ok",
    "data": {
            "pageSize": 10,
            "pageNum": 1,
            "total": 100,
            "list": [{
                        "uid": "1",
                        "validatorIndex": 1234567,
                        "pubkey": "0x1234567890",
                        "balance": "32030000000",
                        "status": "active",
                        "activationTime": "2023/05/04 15:04:05",
                        "exitTime": "",
                        "principal": "32000000000"
                }]
    }
}
```

Node Exit

Used for node exit.

URL: `/validators/{pubkey}/exit`

Method: `POST`

Auth required: YES

Path Parameters:

- pubkey: Pubkey of the node to exit

Parameters:

- epoch: Desired epoch for exit; if not filled, auto exit will be assigned by the program
- broadcast: Whether to broadcast; 0 - No broadcast, 1 - Broadcast on exit

Parameters Example

```
{
        "epoch": 45,
        "broadcast":1,
}
```

Success Response

Code: `200 OK`

Content Definition:

- code: number, 0 is success, others are error
- msg: if response success, it will be "ok", otherwise, it will be error message
- data: If successful, it will return "success".

Content example:

```
{
    "code": 0,
    "msg": "ok",
    "data": "success"
}
```

Earnings-related APIs

Get Daily Earnings Report

Used to retrieve daily earnings report.

URL: `/openapi/rewards/daily`

Method: `GET`

Auth required: YES

Query Parameters:

- uid: User's UID; if not provided or left empty, statistics will be for all users
- startTime: Milliseconds, 00:00:00.000 UTC for the start of the day
- endTime: Milliseconds, 23:59:59.999 UTC for the end of the day

Maximum time range is 1 year.

Query Parameters Example

```
startTime=1682899200000&endTime=1685404799999
```

Success Response

Code: `200 OK`

Content Definition:

- code: number, 0 is success, others are error

- msg: if response success, it will be "ok", otherwise, it will be error message
- data: list of Report Entry

Content example:

```
{
    "code": 0,
    "msg": "ok",
    "data": [
        {
            "date": "20230529",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "3.17%",
            "cumulativeRewards": "2.98339227",
            "cumulativeStaking": "2.52195300",
            "cumulativeGasFee": "0.46143927",
            "rewards": "0.00834306",
            "staking": "0.00834306",
            "gasFee": "0",
            "withdrawal": "0",
            "rate": "1899.65"
        },
        {
            "date": "20230528",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "3.18%",
            "cumulativeRewards": "2.97504921",
            "cumulativeStaking": "2.51360993",
            "cumulativeGasFee": "0.46143927",
            "rewards": "0.00836604",
            "staking": "0.00836604",
            "gasFee": "0",
            "withdrawal": "0.03925047",
            "rate": "1844.17"
        },
        {
            "date": "20230527",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "3.18%",
            "cumulativeRewards": "2.96668316",
            "cumulativeStaking": "2.50524389",
            "cumulativeGasFee": "0.46143927",
            "rewards": "0.00836945",
            "staking": "0.00836945",
            "gasFee": "0",
            "withdrawal": "0",
            "rate": "1828.15"
        }
    ]
}
```

Monthly Report

Used to get monthly statistics report.

URL: `/openapi/rewards/monthly`

Method: `GET`

Auth required: YES

Query Parameters:

- uid: User's UID; if not provided or left empty, statistics will be for all users
- startTime: Milliseconds, 00:00:00.000 UTC for start day of the start month
- endTime: Milliseconds, 23:59:59.999 UTC for end day of the end month

Maximum range is 1 year.

Query Parameters Example

```
startTime=1672531200000&endTime=1680307199999
```

Success Response

Code : `200 OK`

Content Definition:

- code: number, 0 is success, others are error
- msg: if response success, it will be "ok", otherwise, it will be error message
- data: list of Report Entry

Content example

```
{
    "code": 0,
    "msg": "ok",
    "data": [
        {
            "address": "fangzhou@hashquark.io",
            "date": "202303",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "1.01%",
            "cumulativeRewards": "0.55609360",
            "cumulativeStaking": "0.28711542",
            "cumulativeGasFee": "0.26897817",
            "rewards": "0.07454779",
            "staking": "0.06020569",
            "gasFee": "0.01434210",
            "withdrawal": "0",
            "rate": "1792.02"
        },
        {
            "address": "fangzhou@hashquark.io",
            "date": "202302",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "2.23%",
            "cumulativeRewards": "0.48154580",
            "cumulativeStaking": "0.22690973",
            "cumulativeGasFee": "0.25463607",
            "rewards": "0.18204969",
            "staking": "0.09444542",
            "gasFee": "0.08760426",
            "withdrawal": "0",
            "rate": "1622.35"
        },
        {
            "address": "fangzhou@hashquark.io",
            "date": "202301",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "1.62%",
            "cumulativeRewards": "0.29949610",
            "cumulativeStaking": "0.13246430",
            "cumulativeGasFee": "0.16703180",
            "rewards": "0.13263256",
            "staking": "0",
            "gasFee": "0.13263256",
            "withdrawal": "0",
            "rate": "1569.32"
        }
    ]
}
```

Quarterly Report

Used to get quarterly statistics report.

URL: `/openapi/rewards/quarterly`

Method: `GET`

Auth required: YES

Query Parameters:

- uid: User's UID; if not provided or left empty, statistics will be for all users
- startTime: Milliseconds, 00:00:00.000 UTC for start day of the start quarter
- endTime: Milliseconds, 23:59:59.999 UTC for end day of the end quarter

Maximum range is 1 year.

Query Parameters Example

```
startTime=1640995200000&endTime=1672531199999
```

Success Response

Code : `200 OK`

Content Definition:

- code: number, 0 is success, others are error
- msg: if response success, it will be "ok", otherwise, it will be error message
- data: list of Report Entry

Content example

```
{
    "code": 0,
    "msg": "ok",
    "data": [
        {
            "address": "fangzhou@hashquark.io",
            "date": "2022Q4",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "3.80%",
            "cumulativeRewards": "0.94972393",
            "cumulativeStaking": "0.91532469",
            "cumulativeGasFee": "0.03439923",
            "rewards": "0.92071432",
            "staking": "0.88631509",
            "gasFee": "0.03439923",
            "withdrawal": "0",
            "rate": "1196.24"
        },
        {
            "address": "fangzhou@hashquark.io",
            "date": "2022Q3",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "0.11%",
            "cumulativeRewards": "0.02900960",
            "cumulativeStaking": "0.02900960",
            "cumulativeGasFee": "0",
            "rewards": "0.02900960",
            "staking": "0.02900960",
            "gasFee": "0",
            "withdrawal": "0",
            "rate": "1344.91"
        },
        {
            "address": "fangzhou@hashquark.io",
            "date": "2022Q2",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "0.00%",
            "cumulativeRewards": "0",
            "cumulativeStaking": "0",
            "cumulativeGasFee": "0",
            "rewards": "0",
            "staking": "0",
            "gasFee": "0",
            "withdrawal": "0",
            "rate": "1051.37"
        },
        {
            "address": "fangzhou@hashquark.io",
            "date": "2022Q1",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "0.00%",
            "cumulativeRewards": "0",
            "cumulativeStaking": "0",
            "cumulativeGasFee": "0",
            "rewards": "0",
            "staking": "0",
            "gasFee": "0",
            "withdrawal": "0",
            "rate": "3414.39"
        }
    ]
}
```

Annually Report

Used to get annually statistics report. Annually report only support one year each request.

URL: `/openapi/rewards/annually`

Method: `GET`

Auth required: YES

Query Parameters:

- uid: User's UID; if not provided or left empty, statistics will be for all users
- startTime: Milliseconds, 00:00:00.000 UTC for start day of the start year
- endTime: Milliseconds, 23:59:59.999 UTC for end day of the end year

Maximum range is 1 year.

Query Parameters Example

```
startTime=1640995200000&endTime=1672531199999
```

Success Response

Code : `200 OK`

Content Definition:

- code: number, 0 is success, others are error
- msg: if response success, it will be "ok", otherwise, it will be error message
- data: list of Report Entry

Content example

```
{
    "code": 0,
    "msg": "ok",
    "data": [
        {
            "address": "fangzhou@hashquark.io",
            "date": "2022",
            "totalCount": 3,
            "activeCount": 3,
            "exitCount": 0,
            "balance": "96.00957538",
            "principal": "96",
            "apr": "0.98%",
            "cumulativeRewards": "0.94972393",
            "cumulativeStaking": "0.91532469",
            "cumulativeGasFee": "0.03439923",
            "rewards": "0.94972393",
            "staking": "0.91532469",
            "gasFee": "0.03439923",
            "withdrawal": "0",
            "rate": "1196.24"
        }
    ]
}
```

Pooled Staking Related Interfaces

Get Overall Pool Rewards Information

URL: `/pooled_staking/rewards`

Method: `GET`

Authentication Required: YES

Query Parameters:

- date: End date, format `YYYYMMDD` , for example `20230915`
- limit: Display limit

Query Parameters Example

bashCopy code

`date=20230915&limit=10`

Successful Response

Code: `200 OK`

Response Definition:

- code: number, 0 for success, otherwise an error code
- msg: "ok" if successful, otherwise an error message
- data:
  - avg_year_apr: Average annual APR
  - begin_slot: Start slot of the period
  - end_slot: End slot of the period
  - cumulative_reward: Cumulative reward in wei, as a string
  - daily_reward: Daily reward in wei, as a string
  - date: Time in the format "2023-09-19T00:00:21.698+08:00"

Content example:

```
{
  "code": 0,
  "msg": "ok",
  "data": [
    {
      "avg_year_apr": "3.21%",
      "begin_slot": 6502111,
      "cumulative_reward": "19767995899999976",
      "daily_reward": "104072911999999977",
      "date": "2023-09-19T00:00:21.698+08:00",
      "end_slot": 6545312
    }
  ]
}
```

Get Pooled Staking Information

URL: `/pooled_staking/staking_info`

Method: `GET`

Authentication Required: YES

Successful Response

Code: `200 OK`

Response Definition:

- code: number, 0 for success, otherwise an error code
- msg: "ok" if successful, otherwise an error message
- data:
  - amount: Total staked amount in wei, as a string
  - cumulative_reward: Cumulative reward in wei, as a string
  - point: Points in wei, as a string
  - user_address: User's address

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": [{
                "amount": "226917195767195767197",
                "cumulative_reward": "0",
                "point": "226917195767195767197"
        }]
}
```

Get User Staking Information

URL: `/pooled_staking/users/{userAddress}/staking_info`

Method: `GET`

Authentication Required: YES

Path Params

- userAddress: User's address

Note

This endpoint returns the user's staked amount in the entire pool.

Successful Response

Code: `200 OK`

Response Definition:

- code: number, 0 for success, otherwise an error code
- msg: "ok" if successful, otherwise an error message
- data:
  - deposited_eth: Staked ETH amount in wei, as a string
  - total_eth: Current total ETH amount (principal + earnings) in wei, as a string

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "deposited_eth": "291900000000000000000",
                "total_eth": "292090760927820815384"
        }
}
```

Get User Withdrawal Request Information

URL: `/pooled_staking/withdraw_request/{requestID}`

Method: `GET`

Authentication Required: YES

Path Params

- requestID: Withdrawal request ID

Note

This endpoint returns the user's staked amount in the entire pool.

Successful Response

Code: `200 OK`

Response Definition:

- code: number, 0 for success, otherwise an error code
- msg: "ok" if successful, otherwise an error message
- data:
  - point: Total withdrawn points in wei, as a string
  - amount: Total withdrawn ETH amount in wei, as a string
  - principal: ETH principal included in the withdrawal in wei, as a string
  - withdraw_txhash: Transaction Hash, only present if the withdrawal is successful
  - description: Withdrawal status text description

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "request_id": 10,
                "user": "0x8bC4cC7f3eBad85bD3bCf546E2EBdE159D0C39a8",
                "point": "1000000000000000000",
                "amount": "1000000000000000000",
                "principal": "1000000000000000000",
                "withdraw_txhash": "0xd1ecc1a4d80f8c8532880001bd43206c7ffb617db80f1d7901f5a3ecd02773f5",
                "description": "Withdraw finish"
        }
}
```

Get User Withdrawal Possible Block Height

URL: `/pooled_staking/users/{userAddress}/withdraw_possible`

Method: `GET`

Authentication Required: YES

Path Params

- userAddress: User's wallet address

Note

If a user deposits multiple times, this endpoint will return the block height after the last deposit where withdrawals are possible.

Successful Response

Code: `200 OK`

Response Definition:

- code: number, 0 for success, otherwise an error code
- msg: "ok" if successful, otherwise an error message
- data:
  - can_withdraw_block: Withdrawal possible block height

Content example:

```
{
        "code": 0,
        "msg": "ok",
        "data": {
                "can_withdraw_block": "9767771"
        }
}
```

Entities

Assignment

- uid: User's UID, users need to register in advance with the Staking system
- validatorCount: Total number of Validators assigned by the user

UserValidators

- uid: User's UID, users need to register in advance with the Staking system
- validators: List of Validator

Validator

- validatorIndex: Node index
- pubkey: Node public key
- validatorStatus: Node status

ValidatorWithDetails

- validatorIndex: Node index
- pubkey: Node public key
- balance: Current balance
- totalRewards: Total rewards
- status: Current status
- activationTime: Activation time, UTC time, empty string if not activated
- exitTime: Exit time, UTC time, empty string if not exited
- principal: Node principal

UserValidator

- validatorIndex: Node index
- pubkey: Node public key
- validatorStatus: Node status
- uid: User's UID, users need to register in advance with the Staking system

Deposit

- id: Unique identifier
- brokerId: Broker ID
- fromAddress: Deposit address (deprecated)
- depositAddress: Deposit address
- contractAddress: Deposit contract address
- txHash: Hash of the deposit transaction
- chainName: Chain name
- blockHeight: Block height
- slot: Slot count
- blockTime: Transaction block time
- type: Deposit event type, 0 represents broker dimension, 1 represents broker-user dimension
- validators: List of ValidatorWithDetails

ReportEntry

- date: Report date
- totalCount: Total count of validators for all statuses
- activeCount: Count of active validators
- exitCount: Count of exited validators
- balance: Balance of all active validators
- principal: Principal of all active validators
- apr: APR for the period

- cumulativeRewards: Cumulative all rewards until the end of the period
- cumulativeStaking: Cumulative EL rewards until the end of the period
- cumulativeGasFee: Cumulative CL rewards until the end of the period
- rewards: All rewards during the period
- staking: EL rewards during the period
- gasFee: CL rewards during the period
- withdrawal: Total withdrawal amount during the period
- rate: ETH to USD rate until the end of the period

User

- uid: String, user ID, each Broker ensures its uniqueness internally, it's recommended to use the user ID within your system
- email: String, user's email address, optional
- address: String, user's wallet address, optional

UserDetails

- uid: String, user ID, each Broker ensures its uniqueness internally, it's recommended to use the user ID within your system
- email: String, user's email address, optional
- address: String, user's wallet address, optional
- validators: ValidatorWithDetails list
- depositAddresses: deposit address list

ReportEntry

- date:
  - Daily report format: "20230101"
  - Monthly report format: "202301"
  - Quarterly report format: "2023Q1"
  - Annual report format: "2023"
- totalCount: Total count of validators in all states
- activeCount: Number of active validators
- exitCount: Number of exiting validators
- balance: Balance of all active validators
- principal: Principal of all active validators
- apr: Annualized Percentage Rate for the period
- cumulativeRewards: Cumulative rewards until the expiration date
- cumulativeStaking: Cumulative CL rewards until the expiration date
- cumulativeGasFee: Cumulative EL rewards until the expiration date
- rewards: All rewards for the period
- staking: CL rewards for the period
- gasFee: EL rewards for the period
- withdrawal: All withdrawals for the period
- rate: ETH to USD exchange rate until the expiration date

DepositData

- depositDataRoot: Serialized hash of DepositData
- pubkey: Public key
- signature: Signature
- withdrawalCredential: Withdrawal credential