JSON-RPC API

Local Development Server: Accessible at `http://127.0.0.1:4000/api` , this server allows for local testing and development.

Methods

The GenLayer JSON-RPC uses a single endpoint (`/api`). Multiple operations are performed through this endpoint by specifying different "method" values within the request body.

`create_db`

Description: Initializes the database.

Parameters: None

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "create_db",
  "params": [],
  "id": 1
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 1,
  "result": {
    "data": "Database created successfully"
  }
}
```

`create_tables`

Description: Creates necessary tables in the database.

Parameters: None

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "create_tables",
  "params": [],
  "id": 2
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 2,
  "result": {
    "data": "Tables created successfully"
  }
}
```

`create_account`

Description: Registers a new account.

Parameters: None

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "create_account",
  "params": [],
  "id": 3
}
```

Example Response:

```
{
  "jsonrpc": "2.0",
  "id": 3,
  "result": {
    "account_id": "0x12345",
    "message": "Account created successfully"
  }
}
```

`fund_account`

Description: Adds funds to a specified account.

Parameters:

- `address` (string): The account address to fund.
- `amount` (integer): The amount to fund.

Example Request:

```
{
  "jsonrpc": "2.0",
  "method": "fund_account",
  "params": ["0x12345", 100],
  "id": 4
}
```

Example Response:

```
{
  "jsonrpc": "2.0",
  "id": 4,
  "result": {
    "balance": 200,
    "message": "Account funded successfully"
  }
}
```

`send_transaction`

Description: Transfers currency between accounts.

Parameters:

- `from_account` (string): The account sending the currency.
- `to_account` (string): The account receiving the currency.
- `amount` (integer): The amount to transfer.

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "send_transaction",
  "params": {
    "from_account": "0xABC123",
    "to_account": "0xDEF456",
    "amount": 250
  },
  "id": 5
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 5,
  "result": {
    "transaction_id": "0x789abc",
    "message": "Transaction successful"
  }
}
```

`deploy_intelligent_contract`

Description: Deploys a new contract.

Parameters:

- `from_account` (string): The account deploying the contract.
- `contract_code` (string): The path to the contract code.
- `initial_state` (string): The initial state of the contract.
- `class_name` (string): The class name of the contract.

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "deploy_intelligent_contract",
  "params": ["0xABC123", "path/to/contract code", "{}", "MyContract"],
  "id": 6
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 6,
  "result": {
    "contract_address": "0xDef456",
    "message": "Contract deployed successfully"
  }
}
```

`call_contract_function`

Description: Executes a function on a deployed contract.

Parameters:

- `from_account` (string): The account calling the function.
- `contract_address` (string): The contract address.
- `function` (string): The function to call.
- `args` (array): The arguments for the function.

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "call_contract_function",
  "params": ["0xABC123", "0xDEF456", "increaseCount", ["10"]],
  "id": 7
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 7,
  "result": {
    "output": "Function output here",
    "message": "Function called successfully"
  }
}
```

`count_validators`

Description: Counts the number of active validators.

Parameters: None

Example Request:

```json
{
  "jsonrpc": "2.0",
  "method": "count_validators",
  "params": [],
  "id": 8
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 8,
  "result": {
    "count": 10,
    "message": "Validators counted successfully"
  }
}
```

`create_random_validators`

Description: Adds a specified number of validators with random stakes.

Parameters:

- `count` (integer): Number of validators to create.
- `min_stake` (integer): Minimum stake for a validator.
- `max_stake` (integer): Maximum stake for a validator.

Example Request:

```
{
  "jsonrpc": "2.0",
  "method": "create_random_validators",
  "params": [5, 100, 500],
  "id": 9
}
```

Example Response:

```
{
  "jsonrpc": "2.0",
  "id": 9,
  "result": {
    "validators": [
      {"id": "validator1", "stake": 150},
      {"id": "validator2", "stake": 200}
    ],
    "message": "Validators created successfully"
  }
}
```

`delete_all_validators`

Description: Removes all validators.

Parameters: None

Example Request:

```
{
  "jsonrpc": "2.0",
  "method": "delete_all_validators",
  "params": [],
  "id": 10
}
```

Example Response:

```
{
  "jsonrpc": "2.0",
  "id": 10,
  "result": {
    "message": "All validators deleted successfully"
  }
}
```

get_last_contracts

Description: Retrieves the most recent contracts deployed.

Parameters:

- `count` (integer): Number of contracts to retrieve.

Example Request:

```
{
  "jsonrpc": "2.0",
  "method": "get_last_contracts",
  "params": [5],
  "id": 11
}
```

Example Response:

```json
{
  "jsonrpc": "2.0",
  "id": 11,
  "result": [
    {"contract_address": "0xContractAddress1", "deploy_time": "2021-07-20T15:04:05Z"
    {"contract_address": "0xContractAddress2", "deploy_time": "2021-07-19T12:00:00Z"
  ],
  "message": "Last contracts retrieved successfully"
}
```

Last updated on June 19, 2024