**RESTful API**

**The Enara Project**

| **Function Group** | **Type of Request** | **URL** | **Description** |
| --- | --- | --- | --- |
| User | GET | /v1/user/login | Login via GET (only accept SSL) |
| POST | /v1/user/login | Login via POST // {two-factor} to be introduced |
| POST | /v1/user/adduser | Add a new user to the DB |
| POST | /v1/user/deleteuser | Delete a user from the DB |
| POST | /v1/user/changefullname | Change Full Name for {username} |
| GET/POST | /v1/user/logout/{token} | Logout |
| POST | /v1/user/changepassword | Change password for {username} |
| POST | /v1/user/changeusername | Change username for {username} |
| POST | /v1/user/permissions | Change user permissions (0 = No Access / 1 = Read / 2 = Write / 3 = RW) |
|  |  |  |  |
| Configuration | GET | /v1/config | Get all configuration parameters |
| GET | /v1/config/{id} | Get configuration parameter {id} |
| POST | /v1/config | Bulk update of configuration parameters |
| PUT | /v1/config/{id} | Update {id} configuration parameter |
| DELETE | /v1/config/{id} | Delete {id} configuration parameter (only if null config parameter) |
|  |  |  |  |
| Server | GET | /v1/server/status/{token} | Get current status of server components |
| GET | /v1/server/restart/{token} | Restart the node server |
| GET | /v1/server/shutdown/{token} | Shutdown the node server |
| GET | /v1/server/start/{token} | Start the node server |
| GET | /v1/server/generation/hash | Get Generation Nodelist hash |
| GET | /v1/server/generation/key | Get Generation Node Key |
| GET | /v1/server/generation/nodelist | Get Generation Nodelist (limit 1000) |
| Block Chain | GET | /v1/blockchain/hash | Get the current blockchain hash |
| GET | /v1/blockchain/block/{block\_id} | Get the block details and transactions for {block\_id} |
| POST | /v1/blockchain/block/all | Get the block details for {range} of blocks and/or max number {num} |
| GET | /v1/blockchain/transaction/{tx\_id} | Get the Transaction {tx\_id} |
| GET | /v1/blockchain/foundation/hash/{block\_id} | Get the block hash details from the foundation ledger {block\_id} |
| GET | /v1/blockchain/explorer | Simple Web interface to explore the block chain |
|  |  |  |  |
| Node / Peer | GET | /v1/peer/nodelist | Get all nodes that the peer is aware of (limit 1000) |
| GET | /v1/peer/nodelist/bootstrap | Get bootstrap nodelist (random 25) |
| GET | /v1/peer/nodelist/{num} | Get {num} random number of nodes from nodelist (ordered by PING) |
| GET | /v1/peer/nodelist/id/{peer\_id} | Get details for the peer {peer\_id} |
| GET | /v1/peer/challenge | Challenge request for peer |
| GET | /v1/peer/time | Get current Peer Time |
| GET | /v1/peer/gateway | Get gateway mode for Peer |
| GET | /v1/peer/reputation/id/{peer\_id} | Get reputation for {peer\_id} |
| POST | /v1/peer/connect | Connect to Peer |
| GET | /v1/peer/disconnect/{peer\_id} | Disconnect from Peer |
| GET | /v1/peer/ping/{time} | Ping peer |
|  |  |  |  |
| TX Queue | GET | /v1/queue/hash | Get current TX queue Hash |
| GET | /v1/queue/transactions | Get all transactions in queue (limit 1000) |
| POST | /v1/queue/transactions | Broadcast my queued transactions to peer |
| PUT | /v1/queue/transactions/{tx\_id} | Send peer transaction {tx\_id} |
| GET | /v1/queue/transactions/{tx\_id} | Request transaction {tx\_id} from peer |
| Network  Messaging | POST | /v1/message/send | Send a network message via the network |
| GET | /v1/message/queue | Retrieve network messages stored in node queue |
| GET | /v1/message/queue/{id} | Retrieve message {id} from node |
|  |  |  |  |
| Client | GET | /v1/client/balance/hash | Get public key balances hash |
| GET | /v1/client/balance/all | Get all public key balances |
| GET | /v1/client/balance/{pk\_hash} | Get balance for {pk\_hash} |
| POST | /v1/client/connect | Connect to Peer as a client |
| GET | /v1/client/disconnect/{peer\_id} | Disconnect client from Peer |
| GET | /v1/client/ping/{time} | Ping peer as a client |
|  |  |  |  |
| Client  Messaging | POST | /v1/client/message/send | Send a network message via the network |
| GET | /v1/client/message | Retrieve client messages stored at node |
| GET | /v1/client/message/{id} | Retrieve client message {id} from node |