

Connecting with Node.js

This guide explains how to establish a connection between a Node.js application and a Redis database using the [redis](#) package. It walks through the necessary setup, configuration, and execution of a simple Redis command.

Variables

To successfully connect to a Redis instance, you'll need to provide the following parameters. These can typically be found on the Elestio service overview page.

Variable	Description	Purpose
HOST	Redis hostname (from Elestio service overview)	The address of the server hosting your Redis instance.
PORT	Redis port (from Elestio service overview)	The port used for the Redis connection. The default Redis port is 6379.
PASSWORD	Redis password (from Elestio service overview)	Authentication key used to connect securely to the Redis instance.

These values can usually be found in the Elestio service overview details as shown in the image below, make sure to take a copy of these details and add it to the code moving ahead.



redis-aiont

Redis

Cluster

Running

Open terminal

Delete cluster

Add node

Overview

Nodes

Backups

Audit

Termination protection

Disabled. VM can be powered off and terminated.

Protection deactivated



Auto-Failover

Enabled. In case of failure, the cluster will automatically attempt to recover

Auto-Failover activated



Nodes

2 Nodes: 1 Primary, 1 Replica

Add node

Database Admin

Display your database credentials

Hide DB Credentials

Host

redis-aiont-u7774.vm.elestio.app



Port

26379



User

default



Password

Show password



CLI

redis-cli -h redis-aiont-u7774.vm.elestio.app -p 26379 --user default --pass '*****'

Show password



Prerequisites

Install Node.js and NPM

- Check if Node.js is installed by running:

```
node -v
```

- If not installed, download and install it from nodejs.org.
- Confirm npm is installed by running:

```
npm -v
```

Install the redis Package

The redis package enables communication between Node.js applications and Redis.

```
npm install redis --save
```

Code

Create a new file named `redis.js` and add the following code:

```
const redis = require("redis");

// Redis connection configuration
const config = {
  socket: {
    host: "HOST",
    port: PORT,
  },
  password: "PASSWORD",
};

// Create a Redis client
const client = redis.createClient(config);

// Handle connection errors
client.on("error", (err) => {
  console.error("Redis connection error:", err);
});

// Connect and run a test command
(async () => {
  try {
    await client.connect();
    console.log("Connected to Redis");

    // Set and retrieve a test key
    await client.set("testKey", "Hello Redis");
    const value = await client.get("testKey");
    console.log("Retrieved value:", value);

    // Disconnect from Redis
    await client.disconnect();
  } catch (err) {
    console.error("Redis operation failed:", err);
  }
})
```

```
})();
```

To execute the script, open the terminal or command prompt and navigate to the directory where `redis.js` is located. Once in the correct directory, run the script with the command:

```
node redis.js
```

If the connection is successful, the output should resemble:

```
Connected to Redis
```

```
Retrieved value: Hello Redis
```

Revision #2

Created 19 May 2025 06:50:24 by kaiwalya

Updated 19 May 2025 07:28:57 by kaiwalya