How to Connect

Connecting with Node.js

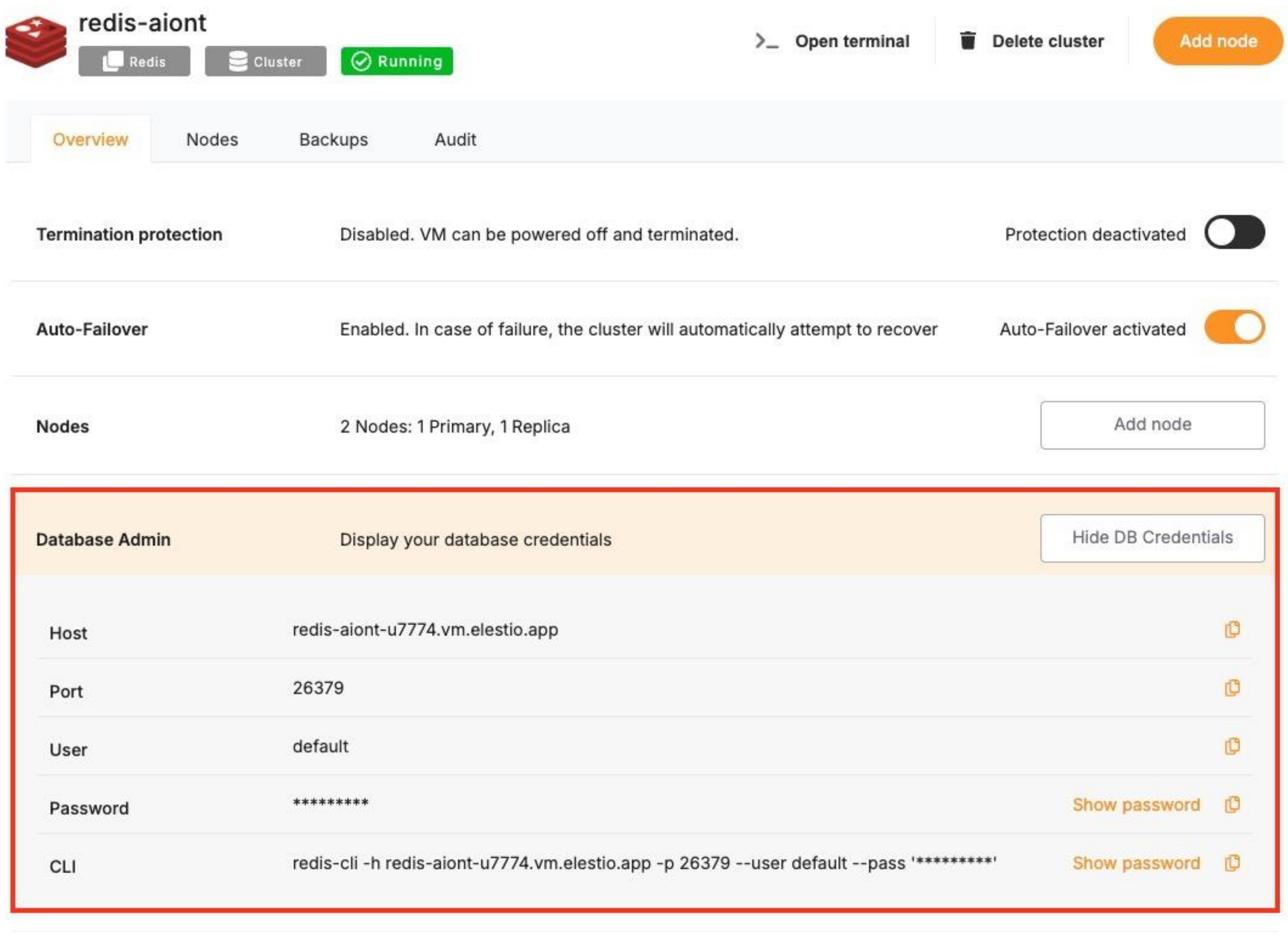
This guide explains how to establish a connection between a Node.js application and a Redis database using the redis package. It walks through the necessary setup, configuration, and execution of a simple Redis command.

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.



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
```

Connecting with Python

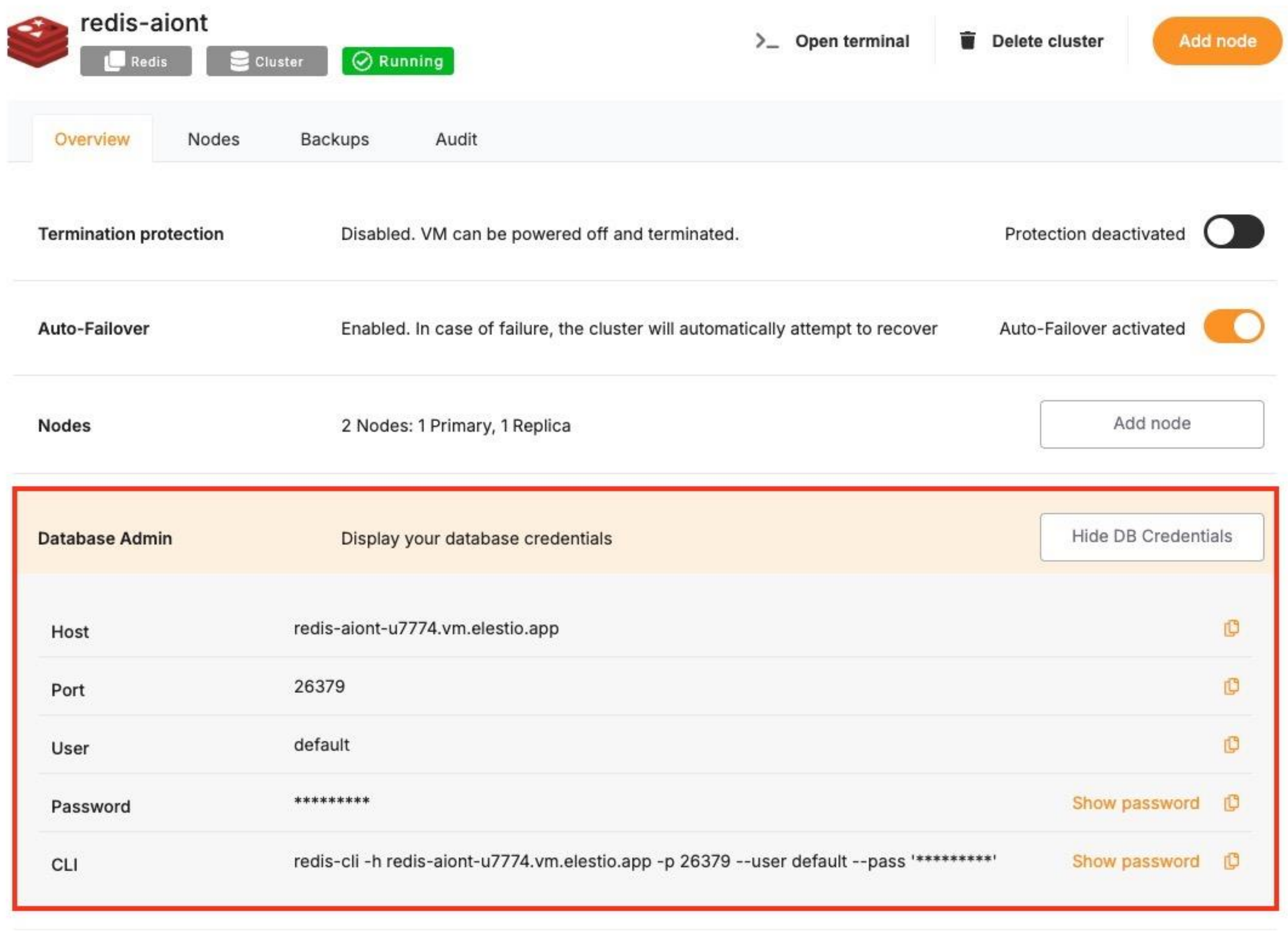
This guide explains how to connect a Python application to a Redis database using the redis library. It walks through the required setup, configuration, and execution of a simple Redis command.

Variables

To connect to Redis, the following parameters are needed. You can find these values in the Elestio Redis service overview.

Variable	Description	Purpose
`HOST`	Redis hostname (from Elestio service overview)	Address of the Redis server.
`PORT`	Redis port (from Elestio service overview)	Port used to connect to Redis. The default is 6379.
`PASSWORD`	Redis password (from Elestio service overview)	Authentication credential for the Redis connection.

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.



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 Python and pip

- Check if Python is installed by running:

```
python3 --version
```

- If not installed, download and install it from python.org.
- Check pip (Python package installer):

```
pip --version
```

Install the redis Package

Install the official redis library using pip:

```
pip install redis
```

Code

Create a file named `redis.py` and paste the following code:

```
import redis


config = {
    "host": "HOST",
    "port": PORT,  # Example: 6379
    "password": "PASSWORD",
    "decode_responses": True
}


try:
    client = redis.Redis(**config)
    client.set("testKey", "Hello Redis")
    value = client.get("testKey")
    print("Connected to Redis")
    print("Retrieved value:", value)


except redis.RedisError as err:
    print("Redis connection or operation failed:", err)
```

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

```
python3 redis.py
```

If everything is set up correctly, the output will be:

```
Connected to Redis
Retrieved value: Hello Redis
```

Connecting with PHP

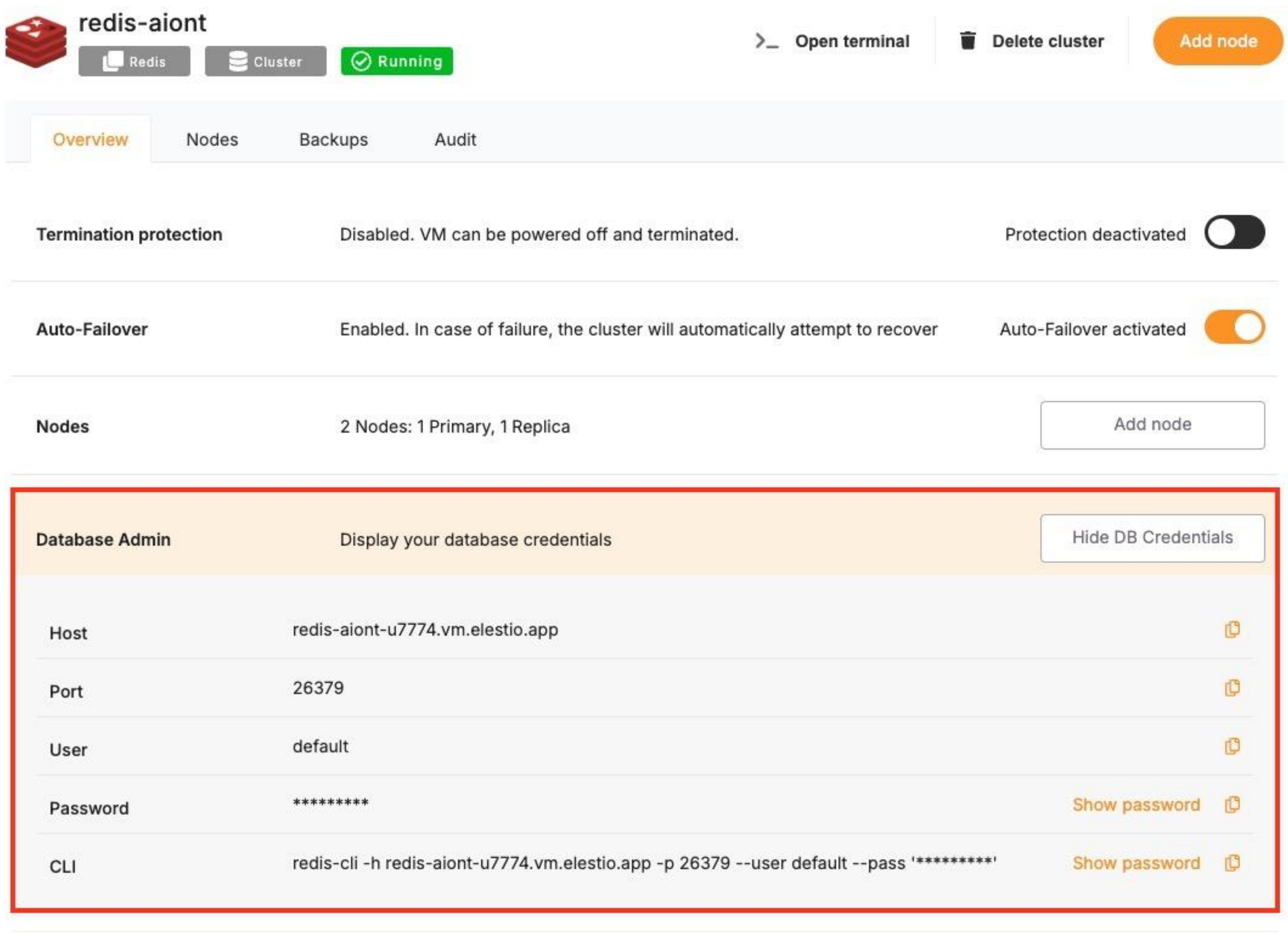
This guide explains how to establish a connection between a PHP application and a Redis database using the phpredis extension. It walks through the necessary setup, configuration, and execution of a simple Redis command.

Variables

Certain parameters must be provided to establish a successful connection to a Redis database. Below is a breakdown of each required variable, its purpose, and where to find it. Here's what each variable represents:

Variable	Description	Purpose
`HOST`	Redis hostname, from the Elestio service overview page	The address of the server hosting your Redis instance.
`PORT`	Port for Redis connection, from the Elestio service overview page	The network port used to connect to Redis. The default port is 6379.
`PASSWORD`	Redis password, from the Elestio service overview page	The authentication key required to connect securely to Redis.

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.



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 PHP**
  - Check if PHP is installed by running:

```
php -v
```

  - If not installed, download it from php.net and install.
- **Install the phpredis Extension**
  - The phpredis extension provides a native PHP interface for Redis. You can install it using:

```
sudo pecl install redis
```

  - Then enable it in your php.ini:

```
extension=redis
```

  - To verify it’s installed:

```
php -m | grep redis
```

Code

Once all prerequisites are set up, create a new file named `redis.php` and add the following code:

```
<?php


$host = 'HOST';
$port = PORT;
$password = 'PASSWORD';


$redis = new Redis();


try {
    $redis->connect($host, $port);


    if (!$redis->auth($password)) {
        throw new Exception('Authentication failed');
    }


    echo "Connected to Redis\n";


    $redis->set("testKey", "Hello Redis");
    $value = $redis->get("testKey");
    echo "Retrieved value: $value\n";


    $redis->close();


} catch (Exception $e) {
    echo "Redis connection or operation failed: " . $e->getMessage() . "\n";
}
```

Open the terminal or command prompt and navigate to the directory where `redis.php` is located. Once in the correct directory, run the script with the command:

```
php redis.php
```

If the connection is successful, the terminal will display output similar to:

```
Connected to Redis
Retrieved value: Hello Redis
```

Connecting with Go

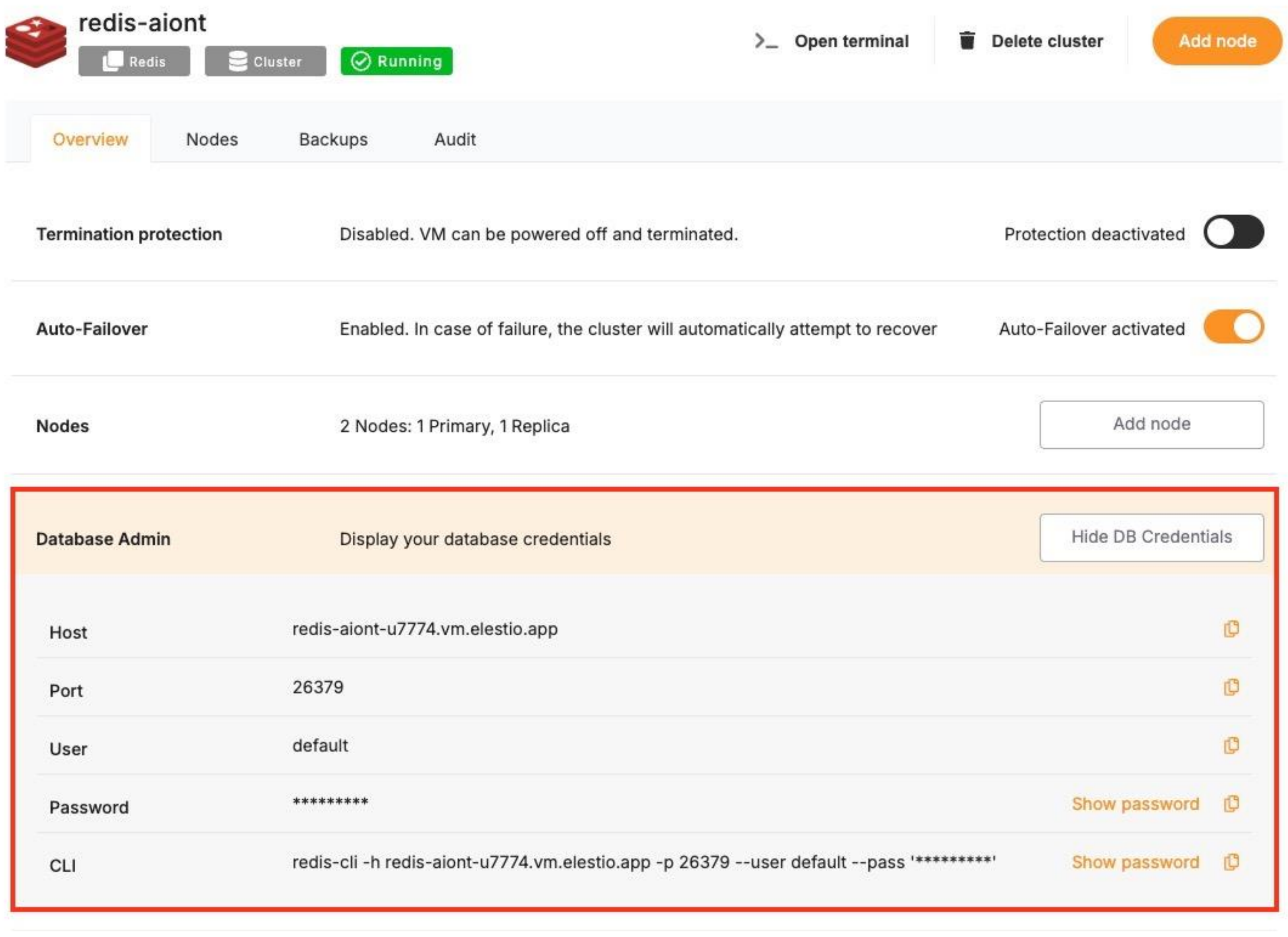
This guide explains how to establish a connection between a Go application and a Redis database using the go-redis package. It walks through the necessary setup, configuration, and execution of a simple Redis command.

Variables

Certain parameters must be provided to establish a successful connection to a Redis database. Below is a breakdown of each required variable, its purpose, and where to find it. Here's what each variable represents:

Variable	Description	Purpose
`HOST`	Redis hostname, from the Elestio service overview page	The address of the server hosting your Redis instance.
`PORT`	Port for Redis connection, from the Elestio service overview page	The network port used to connect to Redis. The default port is 6379.
`PASSWORD`	Redis password, from the Elestio service overview page	The authentication key required to connect securely to Redis.

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.



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

Prerequisites

Install Go

Check if Go is installed by running:

```
go version
```

If not installed, download it from golang.org and install.

Install the go-redis Package

The go-redis package enables Go applications to interact with Redis. Install it using:

```
go get github.com/redis/go-redis/v9
```

Code

Once all prerequisites are set up, create a new file named `redis.go` and add the following code:

```
package main

import (
	"context"
	"fmt"
	"time"

	"github.com/redis/go-redis/v9"
)

func main() {
	opt := &redis.Options{
		Addr:     "HOST:PORT",
		Password: "PASSWORD",
		DB:       0,
	}

	rdb := redis.NewClient(opt)
	ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
	defer cancel()

	err := rdb.Set(ctx, "testKey", "Hello Redis", 0).Err()
	if err != nil {
		fmt.Println("Redis operation failed:", err)
		return
	}

	val, err := rdb.Get(ctx, "testKey").Result()
	if err != nil {
		fmt.Println("Redis operation failed:", err)
		return
	}

	fmt.Println("Connected to Redis")
	fmt.Println("Retrieved value:", val)

	if err := rdb.Close(); err != nil {
		fmt.Println("Error closing connection:", err)
	}
}
```

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

```
go run redis.go
```

If the connection is successful, the terminal will display output similar to:

```
Connected to Redis
Retrieved value: Hello Redis
```

Connecting with Java

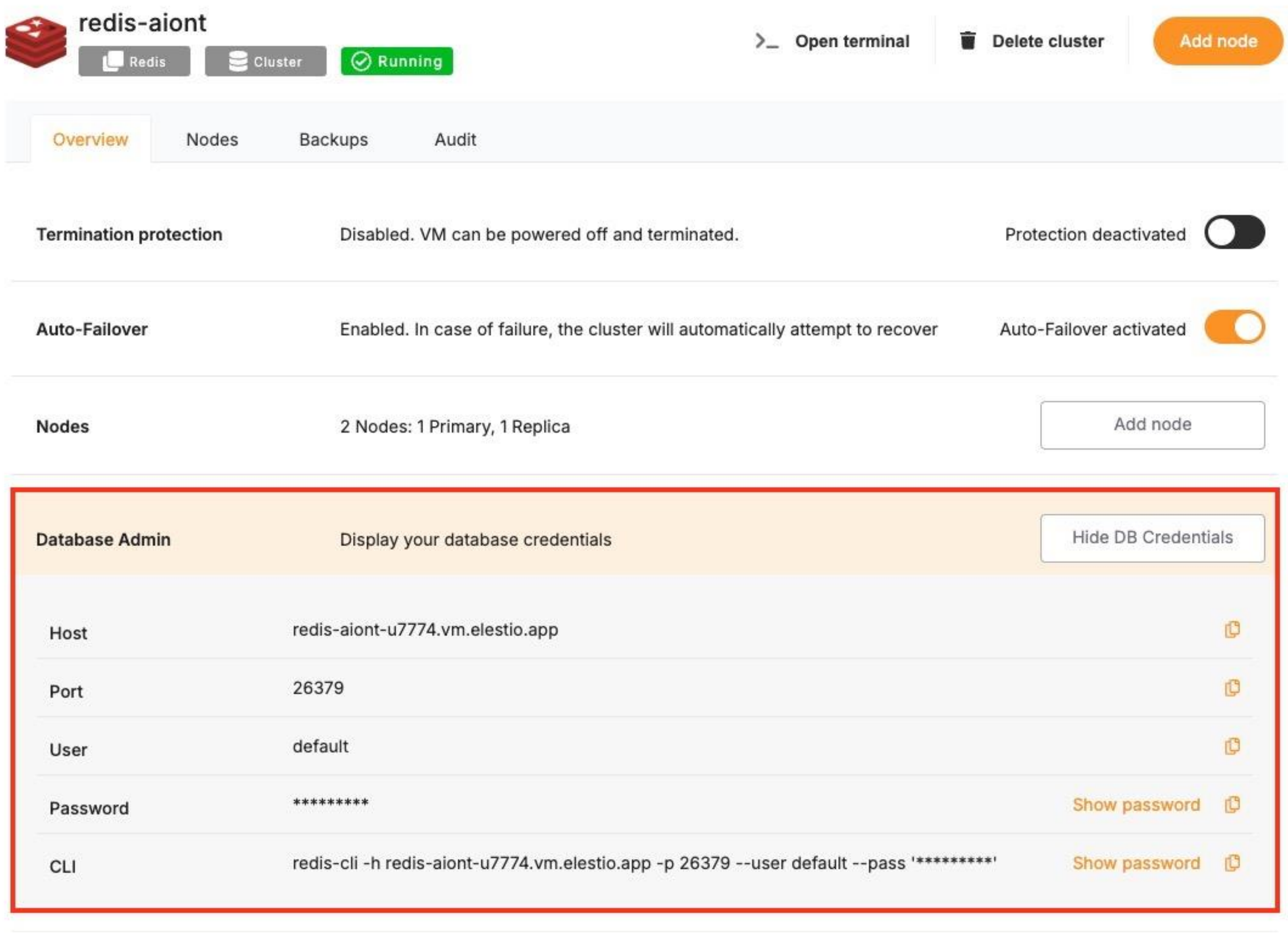
This guide explains how to establish a connection between a Java application and a Redis database using the Jedis library. It walks through the necessary setup, configuration, and execution of a simple Redis command.

Variables

Certain parameters must be provided to establish a successful connection to a Redis database. Below is a breakdown of each required variable, its purpose, and where to find it. Here’s what each variable represents:

Variable	Description	Purpose
`HOST`	Redis hostname, from the Elestio service overview page	The address of the server hosting your Redis instance.
`PORT`	Port for Redis connection, from the Elestio service overview page	The network port used to connect to Redis. The default port is 6379.
`PASSWORD`	Redis password, from the Elestio service overview page	The authentication key required to connect securely to Redis.

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

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 Java

Check if Java is installed by running:

```
java -version
```

If not installed, download it from oracle.com and install.

Download Jedis and Dependencies

The Jedis library enables Java applications to interact with Redis. You need to download two JAR files manually:

1. **Jedis JAR** (Jedis 5.1.0):
   https://repo1.maven.org/maven2/redis/clients/jedis/5.1.0/jedis-5.1.0.jar
2. **Apache Commons Pool2 JAR** (Required by Jedis):

https://repo1.maven.org/maven2/org/apache/commons/commons-pool2/2.11.1/commons-pool2-2.11.1.jar

Place both JAR files in the same directory as your Java file.

Code

Once all prerequisites are set up, create a new file named RedisTest.java and add the following code:

```
import redis.clients.jedis.JedisPooled;

public class RedisTest {
    public static void main(String[] args) {
        // Redis connection configuration
        String host = "HOST";
        int port = PORT; // e.g., 6379
        String password = "PASSWORD";

        // Create a Redis client
        JedisPooled jedis = new JedisPooled(host, port, password);

        try {
            // Set and get a test key
            jedis.set("testKey", "Hello Redis");
            String value = jedis.get("testKey");

            System.out.println("Connected to Redis");
            System.out.println("Retrieved value: " + value);

        } catch (Exception e) {
            System.out.println("Redis connection or operation failed: " + e.getMessage());
        }
    }
}
```

To execute the script, open the terminal or command prompt and navigate to the directory where RedisTest.java is located. Once in the correct directory, run the following commands:

On Linux/macOS :

```
javac -cp "jedis-5.1.0.jar:commons-pool2-2.11.1.jar" RedisTest.java
java -cp ".:jedis-5.1.0.jar:commons-pool2-2.11.1.jar" RedisTest
```

On Windows :

```
javac -cp "jedis-5.1.0.jar;commons-pool2-2.11.1.jar" RedisTest.java
java -cp ".;jedis-5.1.0.jar;commons-pool2-2.11.1.jar" RedisTest
```

If the connection is successful, the terminal will display output similar to:

```
Connected to Redis
Retrieved value: Hello Redis
```

Connecting with RedisInsight

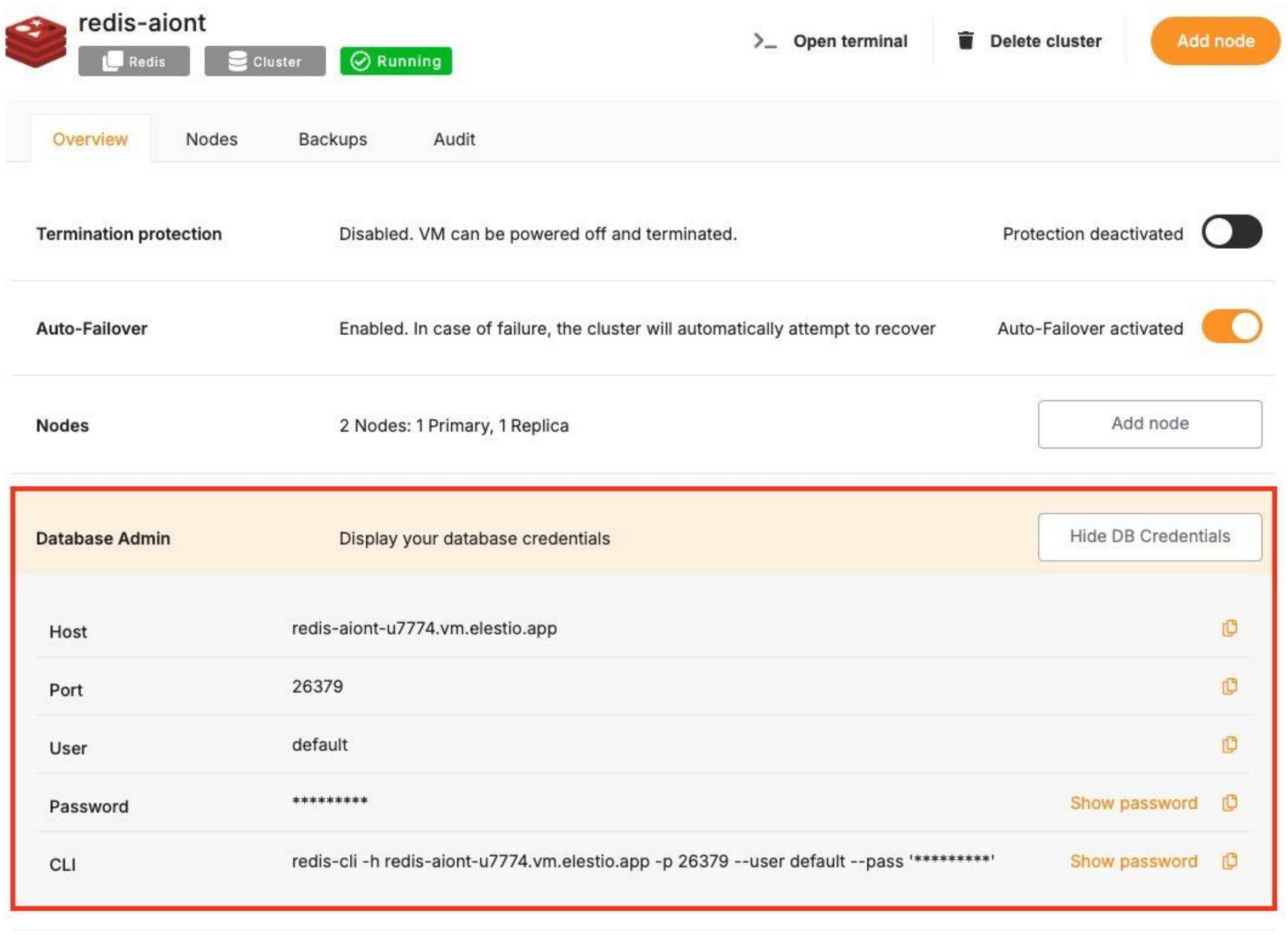
This guide explains how to establish a connection between RedisInsight and a Redis database instance. It walks through the necessary setup, configuration, and connection steps using the official Redis GUI.

Variables

Certain parameters must be provided to establish a successful connection to a Redis database. Below is a breakdown of each required variable, its purpose, and where to find it. Here’s what each variable represents:

Variable	Description	Purpose
`HOST`	Redis hostname, from the Elestio service overview page	The address of the server hosting your Redis instance.
`PORT`	Port for Redis connection, from the Elestio service overview page	The network port used to connect to Redis. The default port is 6379.
`PASSWORD`	Redis password, from the Elestio service overview page	The authentication key required to connect securely to Redis.

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 tool moving ahead.



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 RedisInsight

RedisInsight is a graphical tool for managing Redis databases. Download and install RedisInsight from:

https://redis.com/redis-enterprise/redis-insight/

RedisInsight is available for Windows, macOS, and Linux.

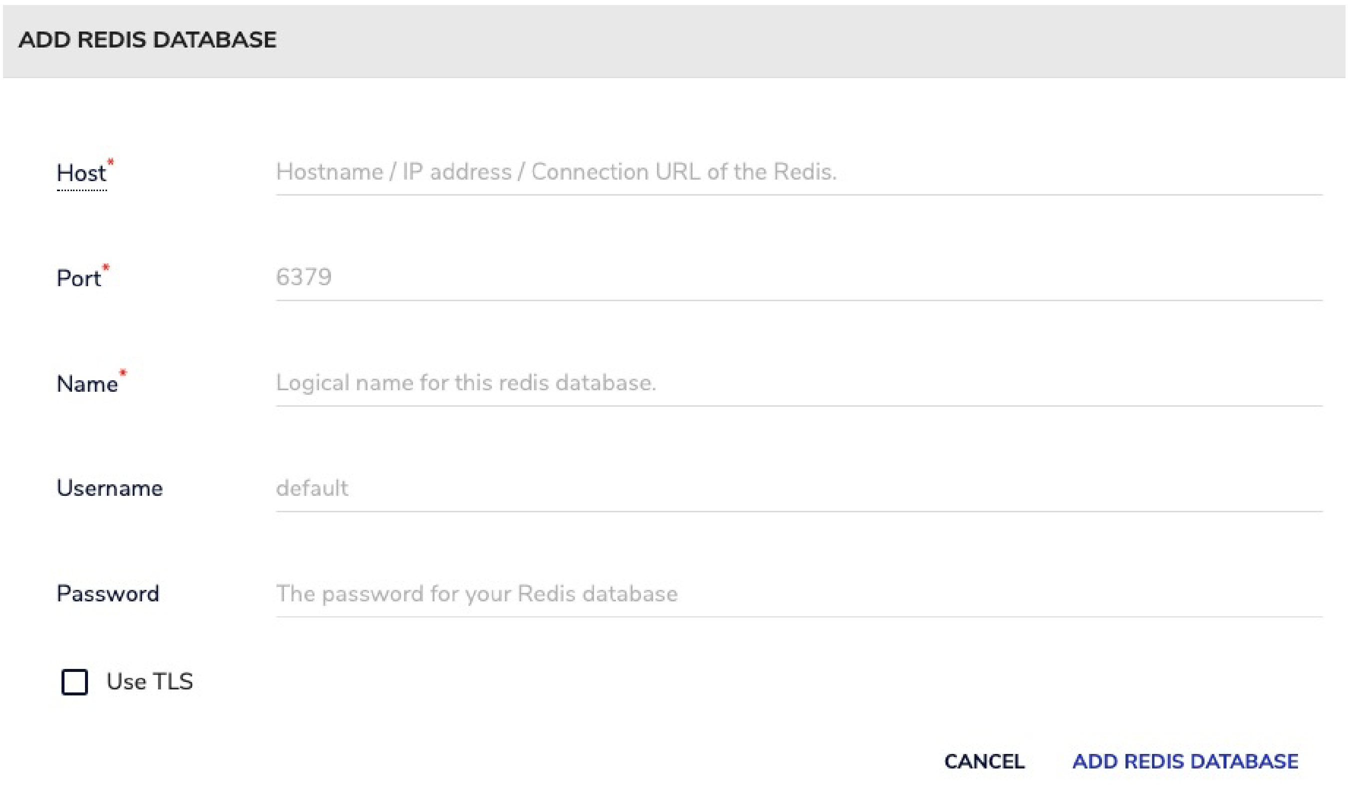
Steps

Once all prerequisites are set up, follow these steps to connect:

1. **Launch RedisInsight**
   Open the RedisInsight application after installation.
2. **Add a New Redis Database**

Click on **“Add Redis Database”**.

3. **Enter Your Connection Details**
   Fill in the following fields using your Elestio Redis service information:
   - **Host**: HOST
   - **Port**: PORT
   - **Password**: PASSWORD



ADD REDIS DATABASE

Host*	Hostname / IP address / Connection URL of the Redis.
Port*	6379
Name*	Logical name for this redis database.
Username	default
Password	The password for your Redis database

☐ Use TLS

4. **Test and Save the Connection**
   Click on **“Test Connection”** to verify the details. If successful, click **“Connect”** or **“Add Database”**.

If the connection is successful, RedisInsight will display a dashboard showing key metrics, data structures, memory usage, and allow you to interact directly with Redis using a built-in CLI or visual browser.

Connecting with redis-cli

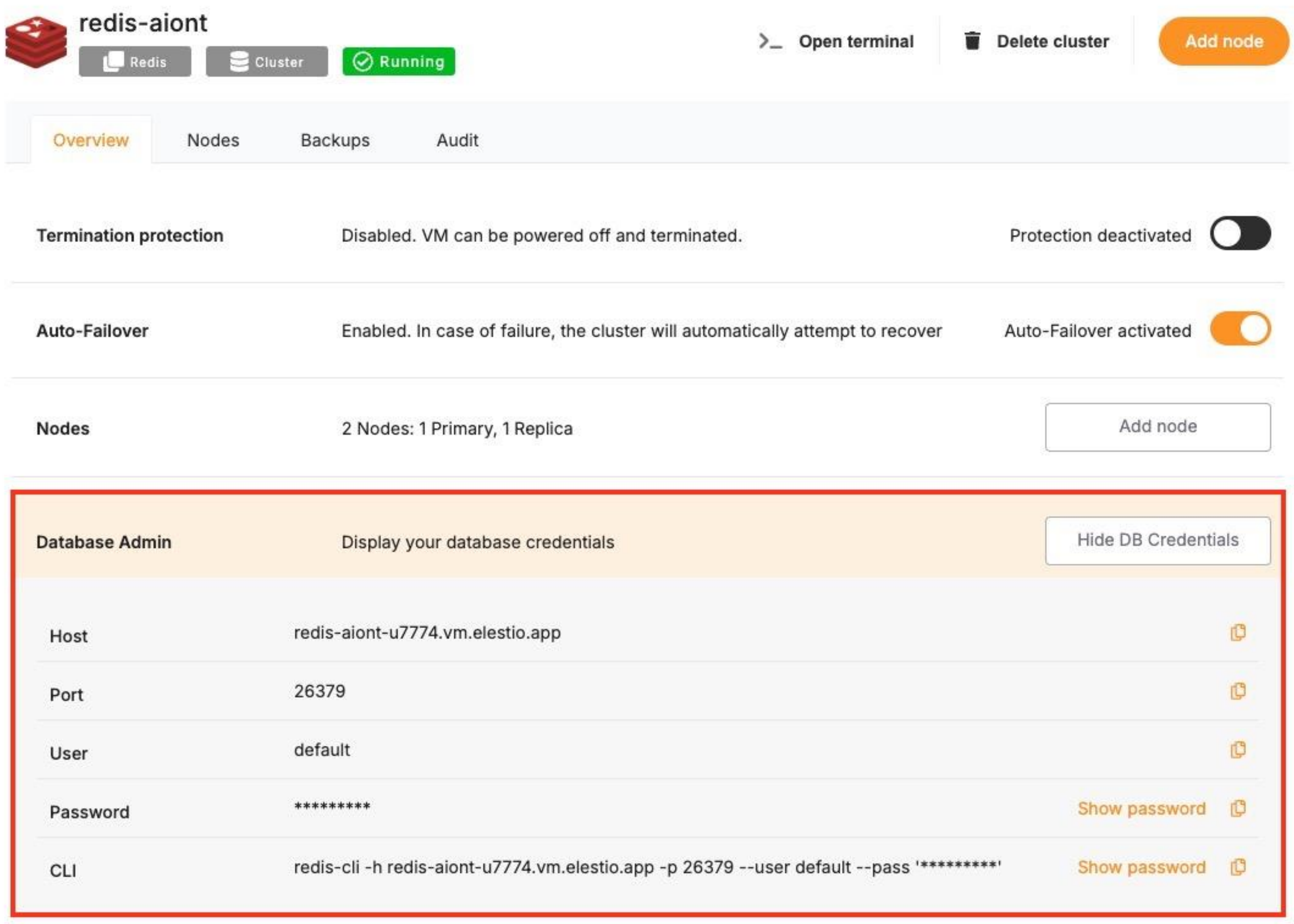
This guide explains how to establish a connection between redis-cli and a Redis database instance. It walks through the necessary setup, configuration, and execution of a simple Redis command from the terminal.

Variables

Certain parameters must be provided to establish a successful connection to a Redis database. Below is a breakdown of each required variable, its purpose, and where to find it. Here’s what each variable represents:

Variable	Description	Purpose
`HOST`	Redis hostname, from the Elestio service overview page	The address of the server hosting your Redis instance.
`PORT`	Port for Redis connection, from the Elestio service overview page	The network port used to connect to Redis. The default port is 6379.
`PASSWORD`	Redis password, from the Elestio service overview page	The authentication key required to connect securely to Redis.

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 use them in the command moving ahead.



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 redis-cli

Check if redis-cli is installed by running:

```
redis-cli --version
```

If not installed, you can install it via:

- **macOS**:

```
brew install redis
```

- **Ubuntu/Debian**:

```
sudo apt install redis-tools
```

- **Windows**:

Use Windows Subsystem for Linux (WSL) or download a Redis CLI binary.

Command

Once all prerequisites are set up, open the terminal or command prompt and run the following command:

```
redis-cli -h HOST -p PORT -a PASSWORD
```

Replace HOST, PORT, and PASSWORD with the actual values from your Elestio Redis service. If the connection is successful, the terminal will display a Redis prompt like this:

```
HOST:PORT>
```

You can then run a simple command to test the connection:

```
set testKey "Hello Redis"
get testKey
```

Expected output:

```
"Hello Redis"
```

If the connection is successful, the terminal will display output similar to:

```
"Hello Redis"
```