

# Connecting with Node.js

This guide explains how to establish a secure connection between a Node.js application and a Keycloak identity provider using the keycloak-connect middleware. It walks through the necessary setup, configuration, and usage of a protected route that requires authentication.

## Variables

Certain parameters must be provided to integrate a Node.js application with Keycloak. Below is a breakdown of each required variable, its purpose, and where to find it. Here's what each variable represents:

Variable	Description	Purpose
REALM	The realm name from the Keycloak Admin Console	Defines the namespace for authentication and authorization
AUTH_SERVER_URL	The full realm URL from Keycloak (e.g., <a href="https://your-domain/realms/xyz">https://your-domain/realms/xyz</a> )	Used as the OIDC issuer base URL
CLIENT_ID	Client ID from the Keycloak Clients page	Identifies the application in Keycloak
CLIENT_SECRET	Secret for the OIDC client, found in the Credentials tab of the client	Authenticates the Node.js application to Keycloak
REDIRECT_URI	URI where users are redirected after authentication	Ensures Keycloak returns control to your app after login

These values can usually be found in the Keycloak Admin Console under **Clients** and **Realm Settings**. Make sure to copy these details and add them to the code moving ahead.

## Prerequisites

### Install Node.js and NPM

Check if Node.js is installed by running:

```
node -v
```

If not installed, download it from <https://nodejs.org> and install.

Verify NPM installation:

```
npm -v
```

## Install Required Packages

The keycloak-connect package enables Node.js applications to authenticate using Keycloak. Install the required packages using:

```
npm install express express-session keycloak-connect
```

## Code

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

```
const express = require("express");
const session = require("express-session");
const Keycloak = require("keycloak-connect");

const app = express();
const port = process.env.PORT || 3000;

const memoryStore = new session.MemoryStore();

app.use(
  session({
    secret: "supersecret",
    resave: false,
    saveUninitialized: true,
    store: memoryStore,
  })
);

const keycloakConfig = {
  realm: "REALM",
  authServerUrl: "AUTH_SERVER_URL",
  clientId: "CLIENT_ID",
  credentials: {
    secret: "CLIENT_SECRET",
  },
  sslRequired: "external",
```

```
    confidentialPort: 0,  
  };  
  
  const keycloak = new Keycloak({ store: memoryStore }, keycloakConfig);  
  app.use(keycloak.middleware());  
  
  app.get("/", (req, res) => {  
    res.send("Welcome to the public route.");  
  });  
  
  app.get("/protected", keycloak.protect(), (req, res) => {  
    res.send("You have accessed a protected route.");  
  });  
  
  app.get("/logout", (req, res) => {  
    req.logout();  
    res.redirect("/");  
  });  
  
  app.listen(port, () => {  
    console.log(`Server running at http://localhost:${port}`);  
  });
```

Replace the placeholder values (REALM, AUTH\_SERVER\_URL, CLIENT\_ID, and CLIENT\_SECRET) with actual values from your Keycloak server.

# Execution

Open the terminal or command prompt and navigate to the directory where keycloak.js is saved. Once in the correct directory, run the script with the command:

```
node keycloak.js
```

If the connection is successful:

1. Visit <http://localhost:3000> in your browser to access the public route.
2. Visit <http://localhost:3000/protected> to trigger Keycloak authentication.
3. Upon successful login, you'll be redirected back and see protected content.
4. Visit <http://localhost:3000/logout> to log out and end the session.

