Role-Based Access Control (RBAC) - Node.js Example

Project Structure

```
rbac-demo/
■
■■■ server.js
■■■ middleware/
■   ■■■ auth.js
■■■ users.js
```

Install Dependencies

```
npm init -y
npm install express jsonwebtoken bcryptjs
```

users.js

```
module.exports = [
  { id: 1, username: "admin", password: "admin123", role: "Admin" },
  { id: 2, username: "mod", password: "mod123", role: "Moderator" },
  { id: 3, username: "user", password: "user123", role: "User" },
];
```

middleware/auth.js

```
const jwt = require("jsonwebtoken");
const SECRET = "MY_SECRET_KEY";

function authenticateToken(req, res, next) {
  const token = req.headers["authorization"]?.split(" ")[1];
  if (!token) return res.status(401).json({ message: "Access denied" });

  jwt.verify(token, SECRET, (err, user) => {
    if (err) return res.status(403).json({ message: "Invalid token" });
    req.user = user;
    next();
  });
}

function authorizeRoles(...allowedRoles) {
  return (req, res, next) => {
    if (!allowedRoles.includes(req.user.role)) {
      return res.status(403).json({ message: "Forbidden: Role not allowed" });
    }
    next();
  };
}

module.exports = { authenticateToken, authorizeRoles, SECRET };
```

server.js

```
const express = require("express");
const jwt = require("jsonwebtoken");
const users = require("./users");
const { authenticateToken, authorizeRoles, SECRET } = require("./middleware/auth");

const app = express();
app.use(express.json());
app.post("/login", (req, res) => {
  const { username, password } = req.body;
  const user = users.find(u => u.username === username && u.password === password);
  if (!user) return res.status(401).json({ message: "Invalid credentials" });

  const token = jwt.sign({ username: user.username, role: user.role }, SECRET, { expiresIn: "1h" });
  res.json({ token });
});

app.get("/", (req, res) => {
  res.send("Welcome to RBAC Demo!");
});

app.get("/profile", authenticateToken, (req, res) => {
  res.json({ message: `Hello ${req.user.username}`, role: req.user.role });
});

app.get("/admin", authenticateToken, authorizeRoles("Admin"), (req, res) => {
  res.send("Welcome Admin! You can manage everything.");
});

app.get("/moderate", authenticateToken, authorizeRoles("Moderator"), (req, res) => {
  res.send("Welcome Moderator! You can manage content.");
});

app.get("/manage", authenticateToken, authorizeRoles("Admin", "Moderator"), (req, res) => {
  res.send("Admin and Moderator can access this route.");
});

app.listen(3000, () => console.log("Server running on port 3000"));
```

Access Control Summary

```
| Route        | Admin | Moderator | User |
|---------------|:-----:|:----------:|:----:|
| `/`           | ■ | ■ | ■ |
| `/profile`    | ■ | ■ | ■ |
| `/admin`      | ■ | ■ | ■ |
| `/moderate`   | ■ | ■ | ■ |
| `/manage`     | ■ | ■ | ■ |
```