**IP HOME ASSIGNMENT - 2**

**Objective:** Implement an authentication middleware that checks if the user is logged in before accessing protected routes.

**Q2.** Create middleware that checks for a valid user session or token.

If the user is authenticated, allow access to the /dashboard route. Otherwise, redirect them to the /login page.

Test the middleware by adding it to routes that should only be accessible to logged-in users.

Show different pages based on user roles (admin vs regular user).

**1. SERVER.JS**

const express = require('express');

const session = require('express-session');

const path = require('path');

const app = express();

app.use(express.urlencoded({ extended: true }));

app.use(session({

secret: 'your\_secret\_key',

resave: false,

saveUninitialized: true

}));

app.set('view engine', 'ejs');

app.set('views', path.join(\_\_dirname, 'views'));

function checkAuth(req, res, next) {

if (req.session && req.session.user) {

return next();

} else {

return res.redirect('/login');

}

}

function isAdmin(req, res, next) {

if (req.session.user && req.session.user.role === 'admin') {

return next();

} else {

return res.status(403).send('Access Denied: Admins Only');

}

}

function isUser(req, res, next) {

if (req.session.user && req.session.user.role === 'user') {

return next();

} else {

return res.status(403).send('Access Denied: Users Only');

}

}

app.get('/login', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'views', 'login.html'));

});

app.post('/login', (req, res) => {

const { username, role } = req.body;

req.session.user = {

username: username,

role: role

};

if (role === 'admin') {

res.redirect('/admin');

} else {

res.redirect('/user');

}

});

app.get('/admin', checkAuth, isAdmin, (req, res) => {

res.render('admin', { user: req.session.user });

});

app.get('/user', checkAuth, isUser, (req, res) => {

res.render('user', { user: req.session.user });

});

app.get('/logout', (req, res) => {

req.session.destroy(err => {

if (err) {

return res.redirect('/user');

}

res.redirect('/login');

});

});

app.listen(3000, () => {

console.log('Server running on http://localhost:3000');

});

2.AUTHMIDDLEWARE.JS  
const express = require('express');

const session = require('express-session');

const app = express();

app.use(express.urlencoded({ extended: true }));

app.use(session({

secret: 'your\_secret\_key',

resave: false,

saveUninitialized: true

}));

function checkAuth(req, res, next) {

if (req.session && req.session.user) {

return next();

} else {

return res.redirect('/login');

}

}

function isAdmin(req, res, next) {

if (req.session.user.role === 'admin') {

return next();

} else {

return res.status(403).send('Access Denied: Admins Only');

}

}

function isUser(req, res, next) {

if (req.session.user.role === 'user') {

return next();

} else {

return res.status(403).send('Access Denied: Users Only');

}

}

app.get('/login', (req, res) => {

res.sendFile(\_\_dirname + '/views/login.html');

});

app.post('/login', (req, res) => {

const { username, role } = req.body;

if (role !== 'admin' && role !== 'user') {

return res.status(400).send('Invalid role selected.');

}

req.session.user = { username, role };

if (role === 'admin') {

res.redirect('/admin');

} else {

res.redirect('/user');

}

});

app.get('/admin', checkAuth, isAdmin, (req, res) => {

res.send('<h1>Welcome to the Admin Dashboard</h1><a href="/logout">Logout</a>');

});

app.get('/user', checkAuth, isUser, (req, res) => {

res.send('<h1>Welcome to the User Dashboard</h1><a href="/logout">Logout</a>');

});

// Logout route

app.get('/logout', (req, res) => {

req.session.destroy((err) => {

if (err) {

return res.status(500).send('Error logging out');

}

res.redirect('/login');

});

});

app.listen(3000, () => {

console.log('Server running on http://localhost:3000');

});

3. ADMIN.EJS

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Admin Dashboard</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

padding: 20px;

background-color: #f4f4f4;

}

h1 {

color: #333;

}

a {

text-decoration: none;

color: #007bff;

}

</style>

</head>

<body>

<h1>Welcome, <%= user.username %>! You are in the Admin Dashboard</h1>

<p>This page is only accessible by administrators.</p>

<a href="/logout">Logout</a> <!-- Add a logout link here -->

</body>

</html>

4. USER.EJS

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>User Dashboard</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

padding: 20px;

background-color: #f4f4f4;

}

</style>

</head>

<body>

<h1>Welcome, <%= user.username %>! You are in the User Dashboard</h1>

<p>This page is only accessible by regular users.</p>

<a href="/logout">Logout</a> <!-- Add a logout link here -->

</body>

</html>

5. LOGIN.HTML

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 0;

}

.container {

max-width: 400px;

margin: 50px auto;

padding: 20px;

background-color: #fff;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

h1 {

text-align: center;

color: #333;

}

input[type="text"],

select {

width: 100%;

padding: 10px;

margin-bottom: 20px;

border: 1px solid #ccc;

border-radius: 4px;

box-sizing: border-box;

}

</style>

</head>

<body>

<div class="container">

<h1>Login</h1>

<form action="/login" method="POST">

<label for="username">Username:</label>

<input type="text" name="username" id="username" required>

<label for="role">Select Role:</label>

<select name="role" id="role" required>

<option value="user">User</option>

<option value="admin">Admin</option>

</select>

<button type="submit">Login</button>

</form>

</div>

</body>

</html>

6. DASHBOARD.EJS

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Dashboard</title>

</head>

<body>

<h1>Welcome to the Dashboard, <%= user.username %>!</h1>

<p>You are logged in as a <%= user.role %>.</p>

<% if (user.role === 'admin') { %>

<a href="/admin">Go to Admin Dashboard</a>

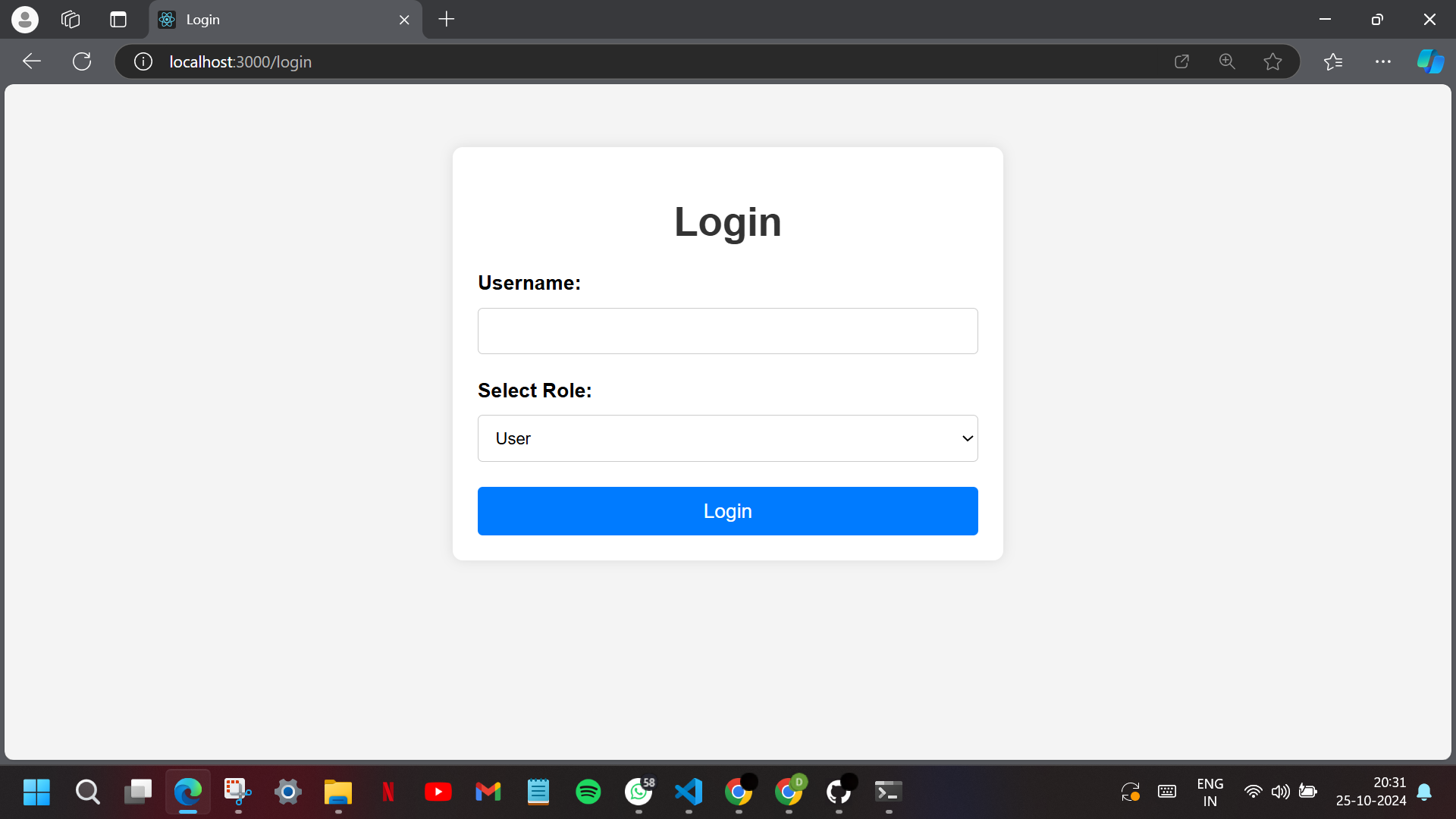
<% } else { %>

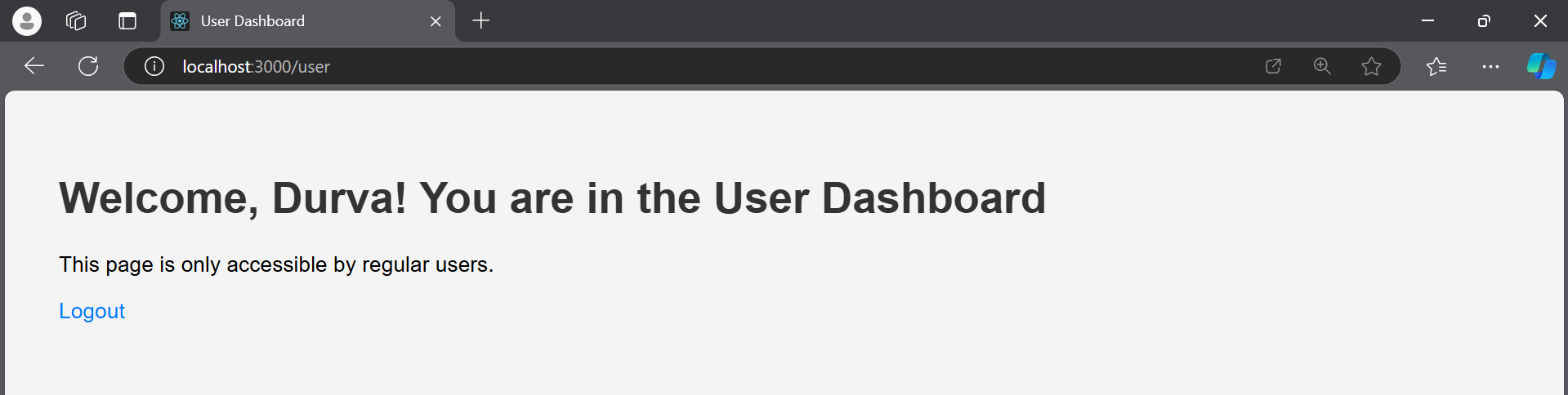
<a href="/user">Go to User Dashboard</a>

<% } %>

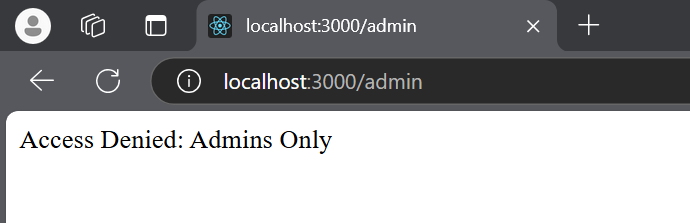
</body>

</html>

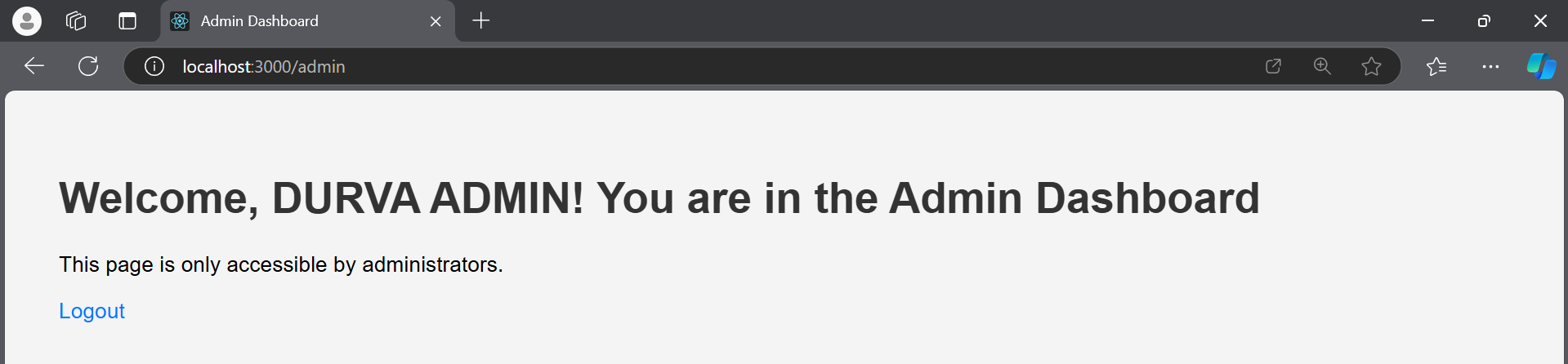
**OUTPUT:**  
LOGIN PAGE:  
  
  
LOGGING IN AS A USER:

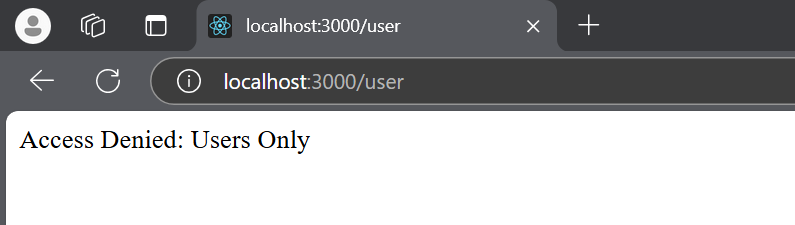


ACCESSING ADMIN PAGE AS A USER:



LOGGING IN AS AN ADMIN:

  
  
ACCESSING USER PAGE AS AN ADMIN:

  
  
**CONCLUSION:** Thus,by utilizing middleware for authentication and authorization, the web application ensures secure access to different dashboards while enhancing the overall user experience.