PS C:\Users\DSLAB\Downloads\leave\_management-main> node -v

v10.15.0

PS C:\Users\DSLAB\Downloads\leave\_management-main> npm --version

6.4.1

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> node .\app.js

C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\mongodb\lib\admin.js:62

session: options?.session,

^

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> npm install

npm WARN read-shrinkwrap This version of npm is compatible with lockfileVersion@1, but package-lock.json was generated for lockfileVersion@3. I'll try to do my best with it!

npm WARN mongodb@6.9.0 requires a peer of socks@^2.7.1 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of snappy@^7.2.2 but none is installed. You must

install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of kerberos@^2.0.1 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of gcp-metadata@^5.2.0 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of @mongodb-js/zstd@^1.1.0 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of mongodb-client-encryption@>=6.0.0 <7 but none

is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of @aws-sdk/credential-providers@^3.188.0 but none is installed. You must install peer dependencies yourself.

npm WARN test@1.0.0 No description

npm WARN test@1.0.0 No repository field.

npm ERR! path C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\.bin\mime.cmd

npm ERR! Refusing to delete C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\.bin\mime.cmd: is outside C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\mime and not a link

npm ERR! File exists: C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\.bin\mime.cmd

npm ERR! Move it away, and try again.

npm ERR! A complete log of this run can be found in:

npm ERR! C:\Users\DSLAB\AppData\Roaming\npm-cache\\_logs\2024-10-16T10\_37\_26\_726Z-debug.log

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> node .\app.js

C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\mongodb\lib\admin.js:62

session: options?.session,

^

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> npm install

npm WARN read-shrinkwrap This version of npm is compatible with lockfileVersion@1, but package-lock.json was generated for lockfileVersion@3. I'll try to do my best with it!

npm WARN mongodb@6.9.0 requires a peer of socks@^2.7.1 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of snappy@^7.2.2 but none is installed. You must

install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of kerberos@^2.0.1 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of gcp-metadata@^5.2.0 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of @mongodb-js/zstd@^1.1.0 but none is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of mongodb-client-encryption@>=6.0.0 <7 but none

is installed. You must install peer dependencies yourself.

npm WARN mongodb@6.9.0 requires a peer of @aws-sdk/credential-providers@^3.188.0 but none is installed. You must install peer dependencies yourself.

npm WARN test@1.0.0 No description

npm WARN test@1.0.0 No repository field.

npm ERR! path C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\.bin\mime.cmd

npm ERR! Refusing to delete C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\.bin\mime.cmd: is outside C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\mime and not a link

npm ERR! File exists: C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\.bin\mime.cmd

npm ERR! Move it away, and try again.

npm ERR! A complete log of this run can be found in:

npm ERR! C:\Users\DSLAB\AppData\Roaming\npm-cache\\_logs\2024-10-16T10\_38\_17\_187Z-debug.log

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> node .\app.js

C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main\node\_modules\mongodb\lib\admin.js:62

session: options?.session,

^

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> npm i

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> node .\app.js

Server running on port 5000

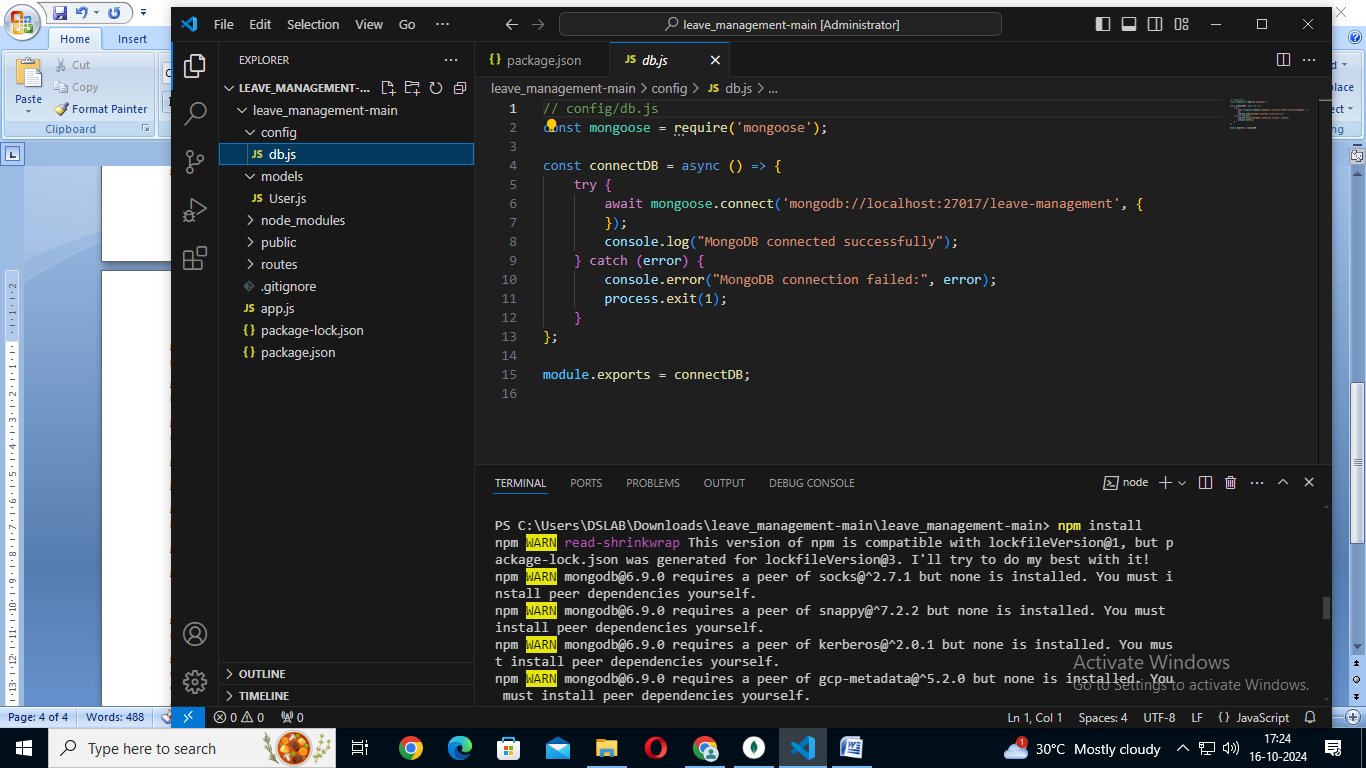
MongoDB connected successfully

PS C:\Users\DSLAB\Downloads\leave\_management-main\leave\_management-main> node .\app.js

Server running on port 5000

MongoDB connected successfully

<https://github.com/Thennavan-Hex/leave_management>



/ config/db.js

const mongoose = require('mongoose');

const connectDB = async () => {

    try {

        await mongoose.connect('mongodb://localhost:27017/leave-management', {

        });

        console.log("MongoDB connected successfully");

    } catch (error) {

        console.error("MongoDB connection failed:", error);

        process.exit(1);

    }

};

module.exports = connectDB;

// models/User.js

const mongoose = require('mongoose');

const UserSchema = new mongoose.Schema({

    username: {

        type: String,

        required: true

    },

    email: {

        type: String,

        required: true,

        unique: true

    },

    rollNumber: {

        type: String,

        required: true

    },

    casualLeave: {

        type: Number,

        default: 10 // default leave balance

    },

    medicalLeave: {

        type: Number,

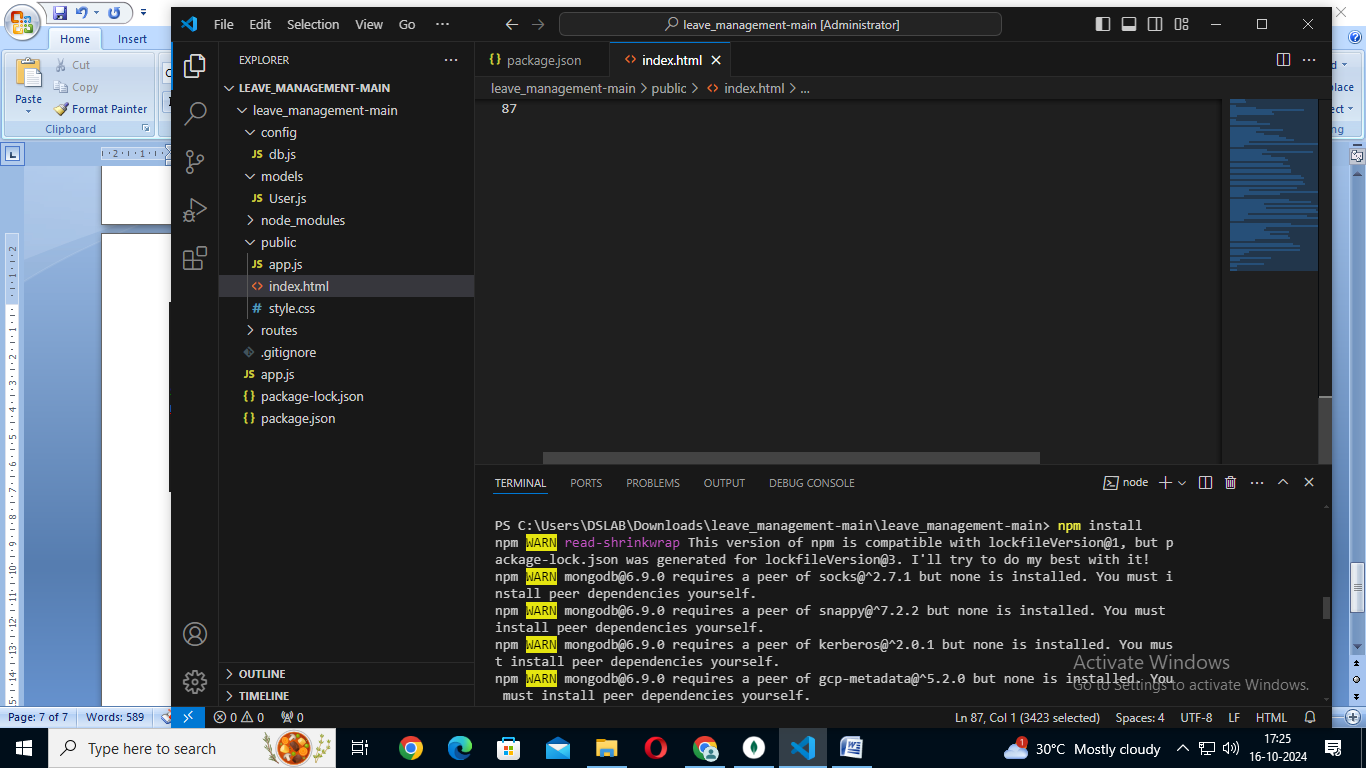
        default: 10 // default leave balance

    }

});

const User = mongoose.model('User', UserSchema);

module.exports = User;



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Leave Management System</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <div class="container">

        <h1>Leave Management System</h1>

        <!-- Login Section -->

        <div id="login-section" class="form-section">

            <h2>Login</h2>

            <form id="login-form">

                <label for="email">Email:</label>

                <input type="email" id="email" required>

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

            </form>

            <p>Don't have an account? <button id="show-registration" class="link-btn">Create User</button></p>

        </div>

        <!-- Registration Section -->

        <div id="registration-section" class="form-section" style="display: none;">

            <h2>Create User</h2>

            <form id="registration-form">

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

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

                <label for="reg-email">Email:</label>

                <input type="email" id="reg-email" required>

                <label for="roll-number">Roll Number:</label>

                <input type="text" id="roll-number" required>

                <label for="casual-leave">Initial Casual Leave:</label>

                <input type="number" id="casual-leave" value="0" required>

                <label for="medical-leave">Initial Medical Leave:</label>

                <input type="number" id="medical-leave" value="0" required>

                <button type="submit" class="btn">Register</button>

            </form>

            <p>Already have an account? <button id="show-login" class="link-btn">Login</button></p>

            <div id="registration-response"></div>

        </div>

        <!-- Leave Balance Section (Hidden until login) -->

        <div class="balance-section" id="balance-section" style="display: none;">

            <h2>Leave Balance</h2>

            <p>Welcome, <span id="username-display"></span> (Roll Number: <span id="roll-number-display"></span>)</p>

            <div id="leave-balance">

                <p>Casual Leave: <span id="casual-leave-display">Loading...</span></p>

                <p>Medical Leave: <span id="medical-leave-display">Loading...</span></p>

            </div>

            <button id="refresh-balance" class="btn">Refresh Balance</button>

            <button id="logout-button" class="btn" style="display: none;">Logout</button>

        </div>

        <!-- Apply for Leave Section (Hidden until login) -->

        <div class="apply-leave-section" id="apply-leave-section" style="display: none;">

            <h2>Apply for Leave</h2>

            <form id="leave-form">

                <label for="leave-type">Leave Type:</label>

                <select id="leave-type">

                    <option value="casual">Casual</option>

                    <option value="medical">Medical</option>

                </select>

                <label for="leave-days">Number of Days:</label>

                <input type="number" id="leave-days" min="1" required>

                <button type="submit" class="btn">Apply Leave</button>

            </form>

        </div>

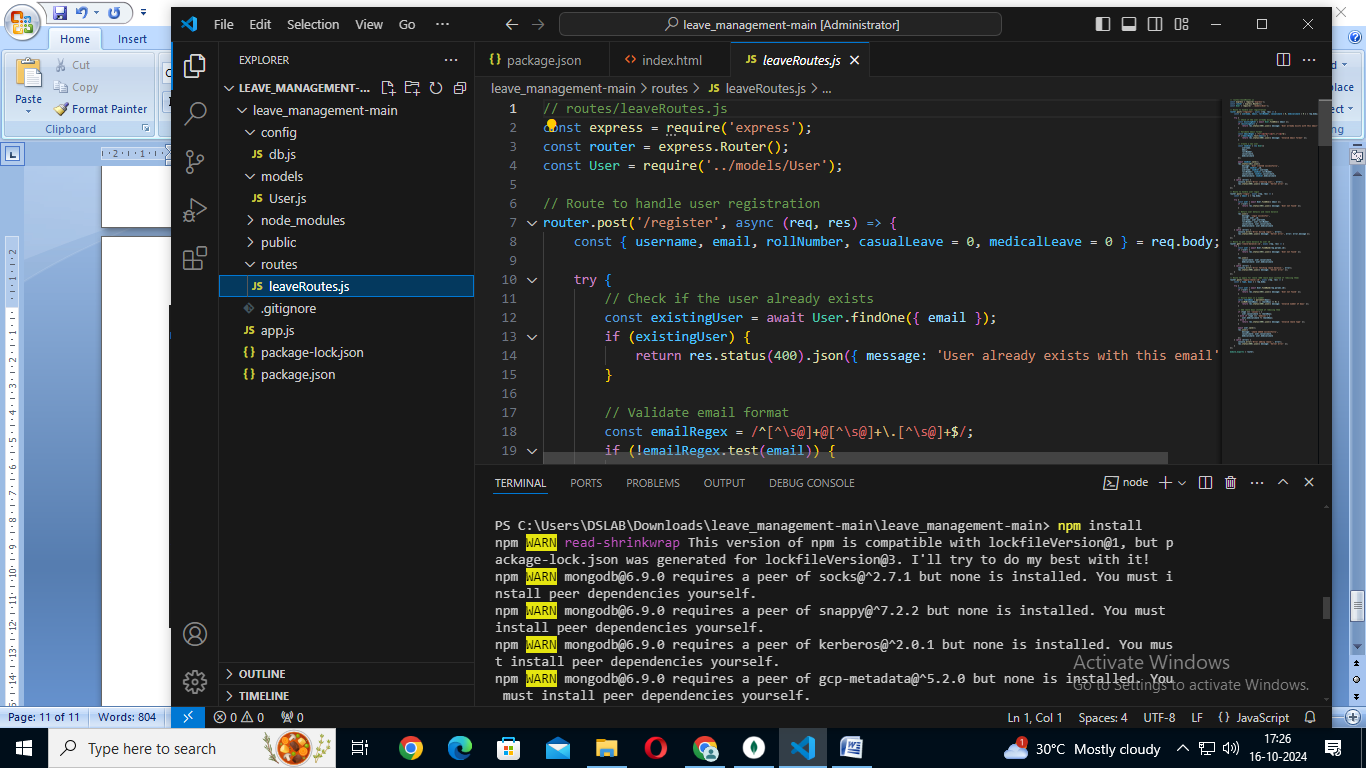
        <div id="response-message"></div>

    </div>

    <script src="app.js"></script>

</body>

</html>



// routes/leaveRoutes.js

const express = require('express');

const router = express.Router();

const User = require('../models/User');

// Route to handle user registration

router.post('/register', async (req, res) => {

    const { username, email, rollNumber, casualLeave = 0, medicalLeave = 0 } = req.body;

    try {

        // Check if the user already exists

        const existingUser = await User.findOne({ email });

        if (existingUser) {

            return res.status(400).json({ message: 'User already exists with this email' });

        }

        // Validate email format

        const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

        if (!emailRegex.test(email)) {

            return res.status(400).json({ message: 'Invalid email format' });

        }

        // Create a new user

        const newUser = new User({

            username,

            email,

            rollNumber,

            casualLeave,

            medicalLeave

        });

        await newUser.save();

        res.status(201).json({

            message: 'User created successfully',

            userId: newUser.\_id,

            username: newUser.username,

            rollNumber: newUser.rollNumber,

            casualLeave: newUser.casualLeave,

            medicalLeave: newUser.medicalLeave

        });

    } catch (error) {

        console.error('Error creating user:', error);

        res.status(500).json({ message: 'Server error' });

    }

});

// Route to handle user login

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

    const { email } = req.body;

    try {

        const user = await User.findOne({ email });

        if (!user) {

            return res.status(404).json({ message: 'User not found' });

        }

        // Return user details and leave balance

        res.json({

            message: 'Login successful',

            userId: user.\_id,

            username: user.username,

            rollNumber: user.rollNumber,

            casualLeave: user.casualLeave,

            medicalLeave: user.medicalLeave

        });

    } catch (error) {

        console.error('Error during login:', error);

        res.status(500).json({ message: 'Server error', error: error.message });

    }

});

// Route to get leave balance by user ID

router.get('/leave-balance/:id', async (req, res) => {

    try {

        const user = await User.findById(req.params.id);

        if (!user) {

            return res.status(404).json({ message: 'User not found' });

        }

        res.json({

            casualLeave: user.casualLeave,

            medicalLeave: user.medicalLeave

        });

    } catch (error) {

        console.error('Error fetching leave balance:', error);

        res.status(500).json({ message: 'Server error' });

    }

});

// Route to apply for leave (ADD leave days instead of reducing them)

router.post('/apply-leave/:id', async (req, res) => {

    const { type, days } = req.body;

    try {

        const user = await User.findById(req.params.id);

        if (!user) {

            return res.status(404).json({ message: 'User not found' });

        }

        // Ensure days is a number

        const leaveDays = parseInt(days);

        if (isNaN(leaveDays) || leaveDays <= 0) {

            return res.status(400).json({ message: 'Invalid number of days' });

        }

        // Add leave days instead of reducing them

        if (type === 'casual') {

            user.casualLeave += leaveDays;

        } else if (type === 'medical') {

            user.medicalLeave += leaveDays;

        } else {

            return res.status(400).json({ message: 'Invalid leave type' });

        }

        await user.save();

        res.json({

            message: 'Leave added successfully',

            casualLeave: user.casualLeave,

            medicalLeave: user.medicalLeave

        });

    } catch (error) {

        console.error('Error adding leave:', error);

        res.status(500).json({ message: 'Server error' });

    }

});

module.exports = router;

