**User Model**  
import mongoose from 'mongoose';

const UserSchema = mongoose.Schema(

{

username:{

type: String,

required: true,

},

password:{

type: String,

required: true,

},

firstname:{

type: String,

required: true,

},

lastname:{

type: String,

required: true,

},

isAdmin:{

type: Boolean,

default: false,

},

},

{timestamps: true}

)

const UserModel = mongoose.model("Users",UserSchema);

export default UserModel;  
  
  
**Auth Route**  
import express from 'express';

import { loginUser, registerUser } from '../Controller/AuthController.js';

const router = express.Router();

router.post('/register',registerUser);

router.post('/login',loginUser);

export default router;  
  
**AuthController**  
import UserModel from "../Models/UserModel.js";

import bcrypt from "bcrypt";

//registering a new user

export const registerUser = async(req,res)=>{

const { firstname, lastname, username, password} = req.body;

const salt = await bcrypt.genSalt(10);

const saltedPassword = await bcrypt.hash(password, salt);

const newUser = new UserModel({firstname, lastname, username, password: saltedPassword});

try{

await newUser.save()

res.status(200).json(newUser)

}

catch(e){

res.status(500).json({message: e.message});

}

};

// login user

export const loginUser = async(req,res)=>{

const {username,password} = req.body;

try{

const user = await UserModel.findOne({username: username})

if(user){

const validity = await bcrypt.compare(password, user.password)

validity? res.status(200).json(user): res.status(400).json("Wrong Password")

}

else{

res.status(400).json("User Does not exist") } }

catch(e){

res.status(500).json({message: e.message});

}

}  
  
**User Controller**  
// UserController.js

import User from '../Models/UserModel.js';

// Get user by id

export const getUserById = async (req, res) => {

try {

const id = req.params.id;

console.log(`User id [${id}]`);

const user = await User.findOne({ \_id: id });

if (user) {

return res.status(200).send({

body: user

});

} else {

return res.status(404).send({

body: "",

message: `User does not exist [id=${id}]`

});

}

} catch (error) {

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

return res.status(500).send({

body: "",

message: `Error in fetching user\n${error}`

});

}

}

// Create a new user

export const createUser = async (req, res) => {

try {

const { username, email, password } = req.body;

console.log(`User [username=${username}, email=${email}, password=${password}]`);

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

if (userFromDB) {

return res.status(400).send({

message: "User already exists"

});

} else {

const newUser = new User({

username,

email,

password

});

const user = await newUser.save();

return res.status(201).send({

body: user,

message: `User created successfully`

});

}

} catch (error) {

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

return res.status(500).send({

body: "",

message: `Error in creating user\n${error}`

});

}

}

// Update user by id

export const updateUser = async (req, res) => {

try {

const { id } = req.params;

const { username, email, password } = req.body;

console.log(`User [id=${id}, username=${username}, email=${email}, password=${password}]`);

const updatedUser = await User.findByIdAndUpdate(id, { username, email, password }, { new: true });

if (updatedUser) {

return res.status(200).send({

body: updatedUser,

message: `User updated successfully`

});

} else {

return res.status(404).send({

body: "",

message: `User not found`

});

}

} catch (error) {

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

return res.status(500).send({

body: "",

message: `Error in updating user\n${error}`

});

}

}

// Delete user by id

export const deleteUser = async (req, res) => {

try {

const { id } = req.params;

const deletedUser = await User.findByIdAndDelete(id);

if (deletedUser) {

return res.status(200).send({

body: "",

message: `User deleted successfully`

});

} else {

return res.status(404).send({

body: "",

message: `User not found`

});

}

} catch (error) {

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

return res.status(500).send({

body: "",

message: `Error in deleting user\n${error}`

});

}

}  
  
**Delete User Route**  
import express from 'express';

import User from '../Models/UserModel.js';

const router = express.Router();

router.delete("/", async (req, res) => {

const { username } = req.body;

try {

const deletedUser = await User.findOneAndDelete({ username: username });

if (!deletedUser) {

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

}

res.json({ message: 'User deleted successfully' });

} catch (err) {

console.error('Error deleting user:', err);

res.status(500).json({ error: 'Internal server error' });

}

});

export default router;  
  
**Update User Route**  
import express from 'express';

import User from '../Models/UserModel.js';

const router = express.Router();

router.put("/:username", async (req, res) => {

const { username } = req.params;

const { email, password } = req.body;

try {

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

if (!user) {

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

}

user.email = email || user.email;

user.password = password || user.password;

const updatedUser = await user.save();

res.json(updatedUser);

} catch (err) {

console.error('Error updating user:', err);

res.status(500).json({ error: 'Internal server error' });

}

});

export default router;  
  
