# Muneeb Hashmi

## Advanced Programming

## Assignment 2

# Code:

## Server.js

require("./models/db");

var User = require('./models/user.model')

const express = require('express')

var bodyParser = require("body-parser")

var multer = require('multer')

var upload = multer()

var session=require('express-session')

var cookieParser = require('cookie-parser')

const userController = require('./controllers/userController')

const productController = require('./controllers/productController')

var app = express()

app.set('view engine','pug')

app.set('views','./views')

app.use(bodyParser.json())

app.use(bodyParser.urlencoded({extended:true}))

app.use(upload.array())

app.use(cookieParser())

app.use(express.static('public'))

app.use(session(

{

secret:'secret key',

resave:true,

saveUninitialized:false

}))

app.get('/', function(req, res)

{

if (req.session.user)

{

res.redirect('/user/protected')

}

else

{

var login = '/user/login';

var signup = '/user';

res.render('index',{login\_url:login, signup\_url:signup })

}

})

app.use('/user', userController)

app.use('/product', productController)

app.listen(3000)

# Controllers:

## UserController.js

const express = require('express')

require("../models/db");

var User = require('../models/user.model')

var bodyParser = require("body-parser")

var multer = require('multer')

var upload = multer()

var session=require('express-session')

var cookieParser = require('cookie-parser')

var router = express.Router();

var login = '/user/login';

var signup = '/user';

var logout = '/user/logout';

var msg = "";

router.get('/', function(req, res)

{

msg ="";

if (!req.session.user)

{

res.render('form',{message:msg, login\_url:login});

}

else

{

console.log("user already logged in")

res.redirect("/");

}

})

router.post('/', function(req, res)

{

var username = req.body.username;

var password = req.body.pass;

var cpassword = req.body.cpass;

var email = req.body.email;

var role = req.body.role;

console.log(username+" "+password+" "+cpassword+" "+email+" "+role);

if (!username || !password || !cpassword || !email)

{

console.log("fields empty");

msg = "Fields are empty. Pls Fill in all fields to proceed the process. Enter the data again";

res.render("form", {message:msg, login\_url:login});

}

else

{

if (password!=cpassword)

{

console.log("Password Not matched");

msg = "Password not matched.Pls enter the data again";

res.render("form", {message:msg, login\_url:login});

}

else

{

User.find({username:username}, function(err, response)

{

if (err) throw err;

console.log(response)

if (!response[0])

{

console.log("unique username found")

User.find({email:email}, function(err, response)

{

if (err) throw err;

if (!response[0])

{

console.log("unique email found")

var newuser = {username:username, password: password, email:email, role:role}

var user\_id;

var user = new User(newuser)

user.save(function (err, User)

{

if (err) throw err;

console.log("user saved in database")

user\_id = User.\_id;

})

req.session.user = {username:username, password: password, email:email, role:role, us\_id:user\_id}

return res.redirect("/user/login");

}

else

{

console.log("No unique email found");

msg = "Email already exists.Pls enter the data again";

res.render("form", {message:msg, login\_url:login});

}

})

}

else

{

console.log("No unique Username found");

msg = "Username is not unique.Pls enter the data again";

res.render("form", {message:msg, login\_url:login});

}

})

}

}

})

router.get('/login', function(req, res)

{

msg = "";

if (req.session.user)

{

console.log("user logged in")

res.redirect('/user/protected')

}

else

{

console.log("user not logged in")

res.render("login",{message:msg, signup\_url:signup})

}

})

router.get('/protected', function(req, res)

{

if (req.session.user)

{

if (req.session.user.role == "admin")

{

console.log("admin logged in")

res.redirect('/user/display')

}

else

{

console.log("user logged in")

res.redirect('/product')

}

}

else

{

console.log("user not logged in")

res.redirect("/user/login")

}

})

router.get('/display', function(req, res)

{

if (req.session.user)

{

if (req.session.user.role == "admin")

{

console.log("admin logged in")

User.find(function(err, response)

{

if (err) throw err;

console.log(response)

console.log(response.length)

var users = response;

res.render("usersdisplay",{request:req.session, logout\_url:logout, users:users})

})

}

else

{

console.log("Only admin can access it")

res.redirect('/user/protected')

}

}

else

{

console.log("user not logged in")

res.redirect("/user/login")

}

})

router.post('/login', function(req, res)

{

var username = req.body.username;

var password = req.body.pass;

if (!username || !password)

{

console.log("fields empty");

msg = "Fields are empty. Pls Fill in all fields to proceed the process. Enter the data again";

res.render("login", {message:msg, signup\_url:signup});

}

else

{

User.find({username:username}, function(err, response)

{

if (err) throw err;

console.log(response[0])

if (response[0])

{

console.log("user exists")

if (response[0].password == password)

{

console.log("password correct")

req.session.user = response[0]

res.redirect("/user/protected")

}

else

{

console.log("Password Wrong");

msg = "Password Incorrect. Enter the data again";

res.render("login", {message:msg, signup\_url:signup});

}

}

else

{

console.log("user doesn't exist")

msg = "Username does not exist. Pls Enter the correct data";

res.render("login", {message:msg, signup\_url:signup});

}

})

}

})

router.get('/logout',function(req,res)

{

req.session.destroy(function(err){

if(err){

console.log(err);

}

else

{

res.redirect('/user/login');

}

})

})

module.exports = router;

## ProductController.js

const express = require('express')

require("../models/db");

var Product = require('../models/product.model')

var bodyParser = require("body-parser")

var multer = require('multer')

var upload = multer()

var session=require('express-session')

var cookieParser = require('cookie-parser')

var login = '/user/login';

var signup = '/user';

var logout = '/user/logout';

var productitems;

var msg ="";

var items;

var router = express.Router();

router.get('/', function(req, res)

{

if (req.session.user)

{

console.log("user logged in")

res.redirect("/product/display")

}

else

{

console.log("user not logged in")

res.redirect("/user/login")

}

})

router.get('/add', function(req, res)

{

if (req.session.user)

{

msg ="";

res.render('additems', {message:msg})

}

else

{

console.log("user not logged in. not authorized");

res.redirect("/");

}

})

router.post('/add', function(req, res)

{

var name = req.body.name;

var price = parseFloat(req.body.price);

var img = "no image" ;

var desc = req.body.desc;

var reviews = req.body.reviews;

var category = req.body.category;

var username = req.session.user.username;

console.log((!name || !price || !category))

if (!name || !price || !category)

{

console.log("fields empty");

msg = "Name, category and price cannot be empty. Pls Fill in all fields to proceed the process. Enter the data again";

res.render("additems", {message:msg});

}

else

{

Product.find({name:name}, function(err, response)

{

if (err) throw err;

console.log(response)

if (!response[0])

{

console.log("unique product found")

var newproduct = {name:name, category:category, desc:desc, price:price,image:img, reviews:reviews, seller\_username:username}

var product = new Product(newproduct)

product.save(function(err, Product)

{

if (err) throw err;

console.log("product saved in database")

console.log(Product)

})

res.redirect("/product/display")

}

else

{

console.log("No unique product found");

msg = "A product can only be added once and by a single seller";

res.render("additems", {message:msg});

}

})

}

})

router.get('/display', function(req, res)

{

if(req.session.user)

{

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

{

Product.find({seller\_username:req.session.user.username}, function(err, response)

{

if (err) throw err;

console.log(response)

console.log(response.length)

productitems = response

res.render("protected",{request:req.session, logout\_url:logout, item:productitems})

})

}

else

{

Product.find(function(err, response)

{

if (err) throw err;

console.log(response)

console.log(response.length)

productitems = response

res.render("protected",{request:req.session, logout\_url:logout, item:productitems})

})

}

}

else

{

console.log("user not logged in. not authorized");

res.redirect("/");

}

})

router.get('/edit/:name', function(req, res)

{

if (req.session.user)

{

if(req.session.user.role=="seller")

{

msg ="";

var pname = req.params.name;

console.log("seller")

console.log(pname)

Product.find({name:pname}, function(err, response)

{

if (err) throw err;

items = response[0]

console.log(items);

res.render("edititems", {message:msg, item:items})

})

}

else

{

console.log("not authorized to go there")

res.redirect('/product')

}

}

else

{

console.log("user not logged in. not authorized");

res.redirect("/");

}

})

router.post('/edit', function(req, res)

{

var name = req.body.name;

var price = parseFloat(req.body.price);

var img = "no image";

var desc = req.body.desc;

var reviews = req.body.reviews;

var category = req.body.category;

console.log((!price || !category))

if (!price || !category)

{

console.log("fields empty");

msg = "Category and price cannot be empty. Pls Fill in all fields to proceed the process. Enter the data again";

res.render("edititems", {message:msg, item:items});

}

else

{

Product.updateOne({name:name},{price:price, image:img,desc:desc, reviews:reviews, category:category}, function(err, response)

{

if (err) throw err;

console.log(response.nModified)

if (response.nModified)

{

console.log("updated");

console.log(response);

}

res.redirect("/product/display")

})

}

})

router.get('/delete/:name', function(req, res)

{

if (req.session.user)

{

if(req.session.user.role=="seller")

{

var pname = req.params.name;

console.log("seller")

console.log(pname)

Product.deleteOne({name:pname}, function(err, response)

{

if (err) throw err;

console.log(response.deletedCount)

if (!response.deletedCount)

{

res.send("error deleting data");

}

res.redirect("/product/display")

})

}

else

{

console.log("not authorized to go there")

res.redirect('/product')

}

}

else

{

console.log("user not logged in. not authorized");

res.redirect("/");

}

})

module.exports=router

# Models

## Db.js

var mongoose = require('mongoose');

mongoose.set('useNewUrlParser', true);

mongoose.set('useUnifiedTopology', true);

mongoose.connect('mongodb://localhost/mydb', function(err)

{

if (err) throw err;

console.log("Mongodb Database my db connected")

});

var User = require('./user.model')

var Product = require('./product.model')

## Product.model.js

const mongoose = require('mongoose');

mongoose.set('useCreateIndex', true);

mongoose.set('useNewUrlParser', true);

mongoose.set('useUnifiedTopology', true);

var productSchema = mongoose.Schema(

{

name:

{

type:String,

required:true,

},

category:

{

type:String,

required:true,

},

desc:

{

type:String,

},

price:

{

type:Number,

required:true

},

image:

{

type:String,

},

reviews:

{

type:String,

},

seller\_username:

{

type:String,

required:true

}

})

var Product = mongoose.model("Product",productSchema);

module.exports = Product;

## User.model.js

const mongoose = require('mongoose');

mongoose.set('useCreateIndex', true);

mongoose.set('useNewUrlParser', true);

mongoose.set('useUnifiedTopology', true);

var userSchema = mongoose.Schema({

username:

{

type:String,

unique:true,

required:true,

trim:true

},

email:

{

type:String,

unique:true,

required:true,

trim:true

},

password:

{

type:String,

required:true,

},

role:

{

type:String,

required:true,

}

})

var User = mongoose.model("User",userSchema)

module.exports = User

# Views

## Userdisplay.pug

html

html

head

title Users of the Product manager

link(rel="stylesheet",type="text/css", href='/style/protected\_style.css')

body

h1 Welcome to TradeGecko

h1 Role: #{request.user.role} Username: #{request.user.username}

a#logout(href=logout\_url) logout?

br

table

thead

th Username

th Password

th Email

th Role

each user in users

tr

td #{user.username}

td #{user.password}

td #{user.email}

td #{user.role}

## Form.pug

html

html

head

title sign up

link(rel="stylesheet",type="text/css", href='/style/form\_style.css')

body

h1 Please Register Yourself!

form(action ="user", method ="POST")

div

label(for = "username") username:

input(type="text", name = "username" value = "Enter Username")

br

div

label(for = "password") password:

input(type="password", name = "pass")

br

div

label(for = "confirmpassword") Confirm password:

input(type="password", name = "cpass")

br

div

label(for = "email") email:

input(type="email", name = "email" value = "Enter Email")

br

div

label(for = "role") role:

select(name="role")

option buyer

option seller

option admin

br

button(type = "submit") Sign up

p #{message}

h2

a(href=login\_url) Login instead?

Br

## Index.pug

html

html

head

title index

link(rel="stylesheet",type="text/css", href='/style/index\_style.css')

body

h1 Welcome to the TradeGecko.

br

br

h1

a(href=login\_url) Sign In

br

h1

a(href=signup\_url) Register

## edititems.pug

html

html

html

head

title edit items

link(rel="stylesheet",type="text/css", href='/style/form\_style.css')

body

h1 Edit Items Here!

form(action ="/product/edit", method ="POST")

div

label(for = "name") name:

input(type="text", name = "name", value = item.name readonly)

br

div

label(for = "category") Category:

input(type="text", name = "category", value = item.category)

br

div

label(for = "price") Price:

input(type="Number", name = "price",step="0.01", value=item.price, min=0)

br

div

label(for = "desc") Description:

textarea(name="desc") #{item.desc}

br

button(type = "submit") Done

p #{message}

## Additems.pug

html

html

head

title Add Items Section

link(rel="stylesheet",type="text/css", href='/style/form\_style.css')

body

h1 Please Enter The Products To Be Added:

form(action ="add", method ="POST")

div

label(for = "name") name:

input(type="text", name = "name")

br

div

label(for = "category") Category:

input(type="text", name = "category")

br

div

label(for = "price") Price:

input(type="Number", name = "price",step="0.01", min=0)

br

div

label(for = "desc") Description:

textarea(name="desc")

br

button(type = "submit") Done

p #{message}

## Login.pug

html

html

head

title login

link(rel="stylesheet",type="text/css", href='/style/form\_style.css')

body

h1 Login Here!

form(action ="login", method ="POST")

div

label(for = "username") username:

input(type="text", name = "username" value = "Enter Username")

br

div

label(for = "password") password:

input(type="password", name = "pass")

br

button(type = "submit") Log in

p #{message}

h2

a(href=signup\_url) Register here?

Br

## Protected.pug

html

html

head

title Product Items list

link(rel="stylesheet",type="text/css", href='/style/protected\_style.css')

body

h1 Welcome to TradeGecko

h1 Role: #{request.user.role} Username: #{request.user.username}

if(request.user.role == 'seller')

a#additems(href='/product/add') add items

a#logout(href=logout\_url) logout?

br

table

thead

th Name

th Category

th Price(Rs)

th description

if(request.user.role == 'seller')

th action

each it in item

tr

td #{it.name}

td #{it.category}

td #{it.price}

td #{it.desc}

if(request.user.role == 'seller')

td

a(href='/product/edit/'+it.name) Edit

a(href='/product/delete/'+it.name) Delete

# Styles

## Form\_styles.css

body

{

font-family: serif;

height: 100%;

text-align: center;

}

h1

{

text-align: center;

margin-top: 50px;

color:cyan;

}

input, select, textarea

{

width: 20%;

padding: 12px 20px;

margin: 10px;

border: 1px solid #ccc;

border-radius: 4px;

align-items: right;

box-sizing: border-box;

}

textarea

{

height: 150px;

border: 2px solid #ccc;

background-color: #f8f8f8;

resize: none;

}

option

{

font-size:1.2em;

}

label

{

font-size:1.5em;

text-transform:capitalize;

color:cadetblue;

font-weight: bold;

margin: 10px;

}

a

{

text-decoration: none;

font-size:1.2em;

text-transform: uppercase;

color:cadetblue;

font-weight: bold;

margin: 10px;

}

button

{

font-size:1.2em;

margin: 10px;

background-color: cadetblue;

border: none;

color: white;

padding: 16px 32px;

text-decoration: none;

cursor: pointer;

}

p

{

font-weight: bold;

color:cadetblue;

font-size:1.2em;

margin: 10px;

}

## Protected\_style.css

body

{

font-family: sans-serif;

height: 100%;

}

h1

{

text-align: center;

margin-top: 30px;

color:cyan;

}

#additems

{

color:lightblue;

font-size:2em;

}

#logout

{

color:lightblue;

position: absolute;

right: 7%;

font-size:2em;

}

a

{

text-decoration: none;

}

table

{

margin-top: 50px;

border-collapse: collapse;

width: 100%;

}

th, td

{

padding: 25px;

text-align: left;

border-bottom: 1px solid #ddd;

}

th

{

height: 40;

}

td

{

width:17%;

}

td a

{

padding: 5px;

font-size:2em;

color:blue;

}

## Index\_style.css

body

{

width:90%;

margin:auto;

font-family: serif;

height: 100%;

background-color: black;

color:white;

}

h1

{

margin-top: 30px;

color:lightblue;

font-weight: bolder;

text-align: center;

}

a

{

text-decoration: none;

font-size:1em;

text-transform: capitalize;

color:cadetblue;

font-weight: bolder;

margin: 10px;

}

# Screenshots:

