Ishmal Tahir

225868

BSCS 7B

Assignment 2

CS 213 Advance Programming (3+1)

Instructor: Dr. Sidra Sultana

CRUD and AMS in Express.Js

**Scenario**

Code Product manager with name, description, price, category, image and reviews where only authentic user (the one who owns the product) can add, update and delete product. Admin can add users and assign roles as seller and buyer.

## Directories:

controllers

controllerP.js

controllerU.js

models

db.js

product.model.js

user.model.js

public

image

def1.png

def2.png

prod1.png

prod2.jpg

prod3.png

prod4.jpg

prod5.png

style

form\_style.css

views

login.pug

form.pug

product.pug

additems.pug

edititems.pug

server.js

## Code:

### controllerP.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) //user logged in*

*{*

*res.redirect("/product/display")*

*}*

*else //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*

*{*

*res.redirect("/");*

*}*

*})*

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

*{*

*var name = req.body.name;*

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

*var image = req.body.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) //incomplete*

*{*

*msg = "Fill out the missing fields. All compulsory data must be entered.";*

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

*}*

*else*

*{*

*if ( /\.(jpe?g|png|gif)$/i.test(image) )*

*{*

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

*{*

*if (err) throw err;*

*console.log(response)*

*if (!response[0])*

*{*

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

*var product = new Product(newproduct)*

*product.save(function(err, Product)*

*{*

*if (err) throw err;*

*console.log(Product)*

*})*

*res.redirect("/product/display")*

*}*

*else*

*{*

*//product not unique*

*msg = "A product can only be added once by a single user";*

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

*}*

*})*

*}*

*else*

*{*

*msg = "Incorrect image extension. Try again.";*

*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;*

*productitems = response*

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

*})*

*}*

*else*

*{*

*Product.find(function(err, response)*

*{*

*if (err) throw err;*

*productitems = response*

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

*})*

*}*

*}*

*else*

*{*

*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;*

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

*{*

*if (err) throw err;*

*items = response[0]*

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

*})*

*}*

*else //not authorized*

*{*

*res.redirect('/product')*

*}*

*}*

*else*

*{*

*res.redirect("/");*

*}*

*})*

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

*{*

*var name = req.body.name;*

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

*var image = req.body.image;*

*var desc = req.body.desc;*

*var reviews = req.body.reviews;*

*var category = req.body.category;*

*if (!price || !category) //empty fields*

*{*

*msg = "Fill out the missing fields. All compulsory data must be entered.";*

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

*}*

*else*

*{*

*if ( /\.(jpe?g|png|gif)$/i.test(image) )*

*{*

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

*{*

*if (err) throw err;*

*if (response.nModified)*

*{*

*console.log("updated");*

*}*

*res.redirect("/product/display")*

*})*

*}*

*else*

*{*

*console.log("incorrect image extension.");*

*msg = "Incorrect image extension. Try Again.";*

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

*}*

*}*

*})*

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

*{*

*if (req.session.user)*

*{*

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

*{*

*var pname = req.params.name;*

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

*{*

*if (err) throw err;*

*if (!response.deletedCount)*

*{*

*res.send("error deleting data")*

*}*

*})*

*}*

*else*

*{*

*res.redirect('/product')*

*}*

*}*

*else*

*{*

*res.redirect("/");*

*}*

*})*

*module.exports=router*

### controllerU.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 //user 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) //empty fields*

*{*

*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)*

*{*

*msg = "Incorrect Password.Please re-enter the data";*

*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]) //unique username*

*{*

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

*{*

*if (err) throw err;*

*if (!response[0]) //unique email*

*{*

*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*

*{``*

*msg = "Email already exists. Please re-enter your data";*

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

*}*

*})*

*}*

*else*

*{*

*msg = "Username already exists. Please re-enter your data";*

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

*}*

*})*

*}*

*}*

*})*

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

*{*

*msg = "";*

*if (req.session.user) //login*

*{*

*res.redirect('/user/product')*

*}*

*else //signup*

*{*

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

*}*

*})*

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

*{*

*if (req.session.user)*

*{*

*res.redirect('/product')*

*}*

*else*

*{*

*res.redirect("/user/login")*

*}*

*})*

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

*{*

*var username = req.body.username;*

*var password = req.body.pass;*

*if (!username || !password) //incomplete*

*{*

*msg = "Fill out the missing fields. All compulsory data must be entered.";*

*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]) //correct user*

*{*

*if (response[0].password == password) //correct password*

*{*

*req.session.user = response[0]*

*res.redirect("/user/product")*

*}*

*else //incorrect password*

*{*

*msg = "Incorrect Password. Try Again";*

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

*}*

*}*

*else //incorrect user*

*{*

*msg = "Incorrect Username. Try Again";*

*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;*

### db.js

*const 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*

### form\_style.css

*body*

*{*

*background-color: peachpuff;*

*padding: 5%;*

*width:90%;*

*margin:auto;*

*font-family: sans-serif;*

*color: brown;*

*}*

*h1*

*{*

*font-weight: bold;*

*}*

*h2{*

*color:indianred;*

*font-weight: bold;*

*}*

*input, select, textarea*

*{*

*width: 40%;*

*padding: 12px 20px;*

*margin: 10px 900px 10px 10px;*

*border: 1px solid #ccc;*

*border-radius: 4px;*

*align-items: right;*

*box-sizing: border-box;*

*}*

*textarea*

*{*

*height: 150px;*

*border: 2px solid #ccc;*

*background-color: sandybrown;*

*resize: none;*

*}*

*option*

*{*

*font-size:1.2em;*

*}*

*label*

*{*

*font-size:1.3em;*

*text-transform:capitalize;*

*color:sandybrown;*

*font-weight: bold;*

*margin: 10px;*

*}*

*a*

*{*

*text-decoration: none;*

*font-size:1.5em;*

*color:peachpuff;*

*background-color: lightcoral;*

*font-weight: bold;*

*}*

*button*

*{*

*font-size:1.2em;*

*margin: 10px;*

*background-color: lightcoral;*

*border: none;*

*color: white;*

*padding: 16px 32px;*

*text-decoration: none;*

*cursor: pointer;*

*}*

*p*

*{*

*font-weight: bold;*

*color:indianred;*

*font-size:1.2em;*

*margin: 10px;*

*}*

*#logout*

*{*

*position: absolute;*

*right: 10%;*

*}*

*#signup*

*{*

*font-size:1.2em;*

*}*

*table*

*{*

*margin-top: 50px;*

*border-collapse: collapse;*

*width: 100%;*

*}*

*th, td*

*{*

*padding: 25px;*

*text-align: center;*

*border-bottom: 1px solid #ddd;*

*}*

*th*

*{*

*height: 50;*

*}*

*td*

*{*

*background-color: white;*

*width:15%;*

*text-transform: capitalize;*

*}*

*td a*

*{*

*padding: 3px;*

*font-weight: bold;*

*font-size: 1.2em;*

*}*

### login.pug

*html*

*head*

*title login*

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

*body*

*h1 Product Management System*

*br*

*h2 Login*

*form(action ="/user/login", method ="POST")*

*div*

*label(for = "username") username:*

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

*br*

*div*

*label(for = "password") password:*

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

*br*

*button(type = "submit") Log in*

*p #{message}*

*br*

*h2 No Account?*

*a#signup(href=signup\_url) Sign Up Now*

*br*

### form.pug

*html*

*head*

*title sign up*

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

*body*

*h1 Create an Account*

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

*div*

*label(for = "username") username:*

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

*br*

*div*

*label(for = "role") role:*

*select(name="role")*

*option buyer*

*option seller*

*br*

*button(type = "submit") Sign up*

*p #{message}*

*br*

*h2*

*a#signup(href=login\_url) Login*

*br*

### product.pug

*html*

*head*

*title Product Items list*

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

*body*

*h1 Product Management System*

*h2 Current User: #{request.user.username}*

*a#logout(href=logout\_url) Logout*

*h2 You are logged in.*

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

*h2 You can Add, Delete, or Edit Products*

*br*

*a#additems(href='/product/add') Add Items*

*br*

*table*

*thead*

*th Product Name*

*th Category*

*th Price(Rs)*

*th Description*

*th Image*

*th Reviews*

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

*th Action*

*each i in item*

*tr*

*td #{i.name}*

*td #{i.category}*

*td #{i.price}*

*td #{i.desc}*

*td*

*image(src='/image/'+i.image, alt=i.image,width="70", height="70")*

*td #{i.reviews}*

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

*td*

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

*br*

*br*

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

### additems.pug

*html*

*head*

*title add items*

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

*body*

*h1 Add Products*

*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 = "image") Image Url:*

*input(type="text", name = "image", value="def1.png")*

*br*

*div*

*label(for = "desc") Description:*

*textarea(name="desc")*

*br*

*div*

*label(for = "reviews") Reviews:*

*textarea(name="reviews")*

*br*

*button(type = "submit") Done*

*p #{message}*

### edititems.pug

*html*

*head*

*title edit items*

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

*body*

*h1 Edit*

*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 = "image") Image Url:*

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

*br*

*div*

*label(for = "desc") Description:*

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

*br*

*div*

*label(for = "reviews") Reviews:*

*textarea(name="reviews") #{item.reviews}*

*br*

*button(type = "submit") Done*

*p #{message}*

### 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 controllerU = require('./controllers/controllerU')*

*const controllerP = require('./controllers/controllerP')*

*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:' s1 ',*

*resave:true,*

*saveUninitialized:false*

*}))*

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

*{*

*if (req.session.user)*

*{*

*res.redirect('/user/product')*

*}*

*else*

*{*

*var login = '/user/login';*

*var signup = '/user';*

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

*}*

*})*

*app.use('/user', controllerU)*

*app.use('/user/login', controllerU)*

*app.use('/user/product', controllerU)*

*app.use('/product', controllerP)*

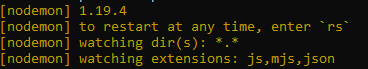
*app.use('/product/edit/:name', controllerP)*

*app.use('/user/logout', controllerU)*

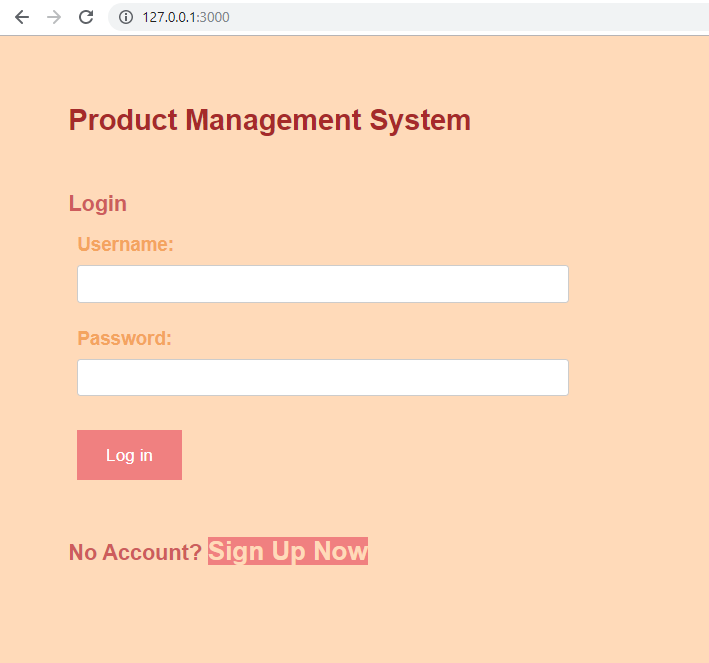
*app.listen(3000)*

## Output:

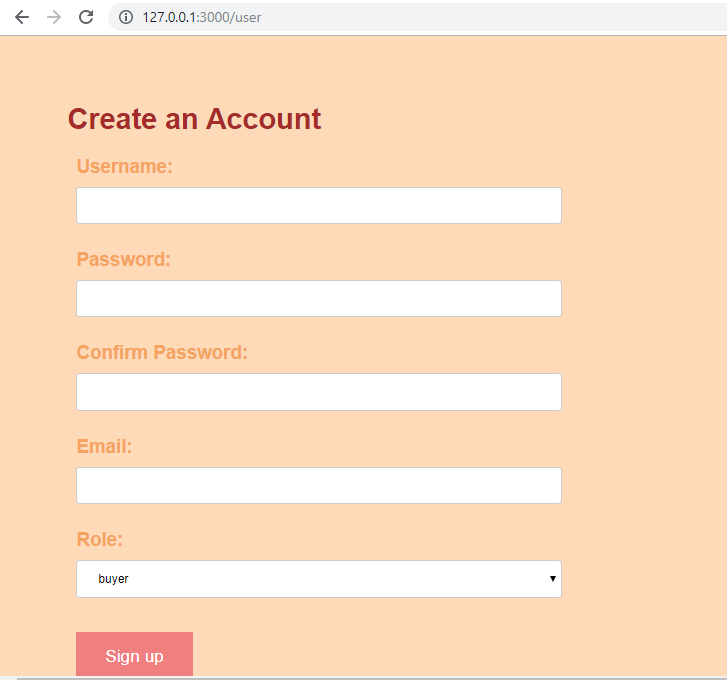
#### Start nodemon server



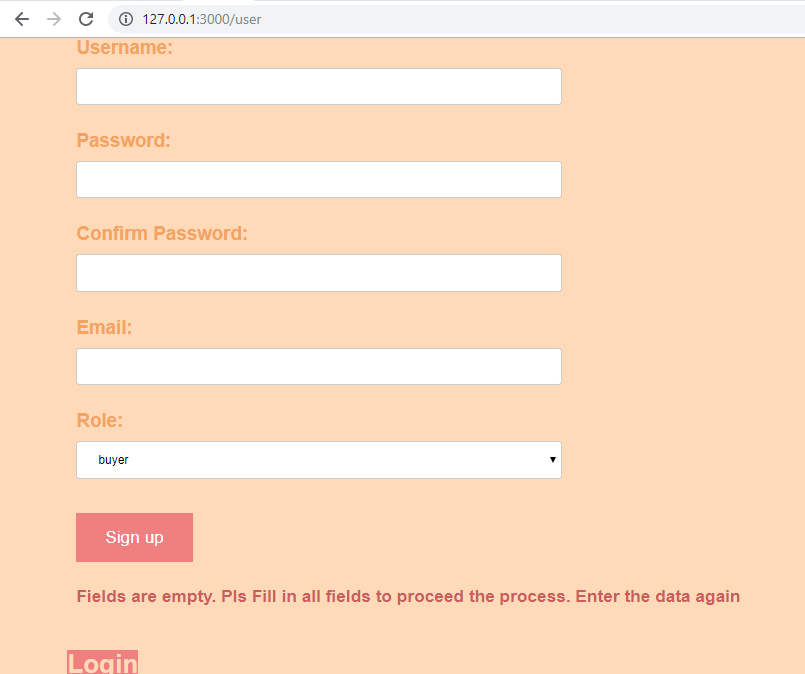
#### Open the browser and enter the port address. Login page will be displayed.



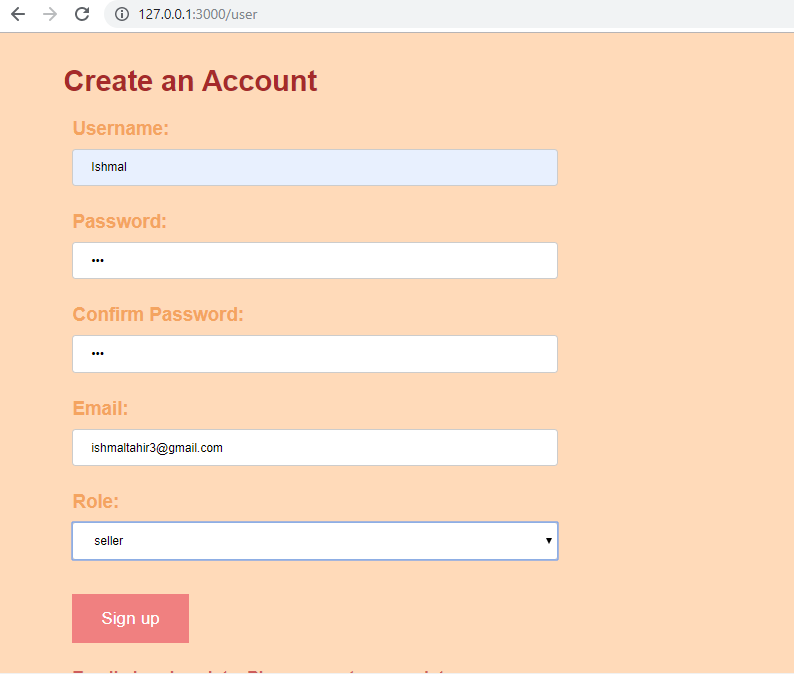
#### If you do not already have an account, create one.



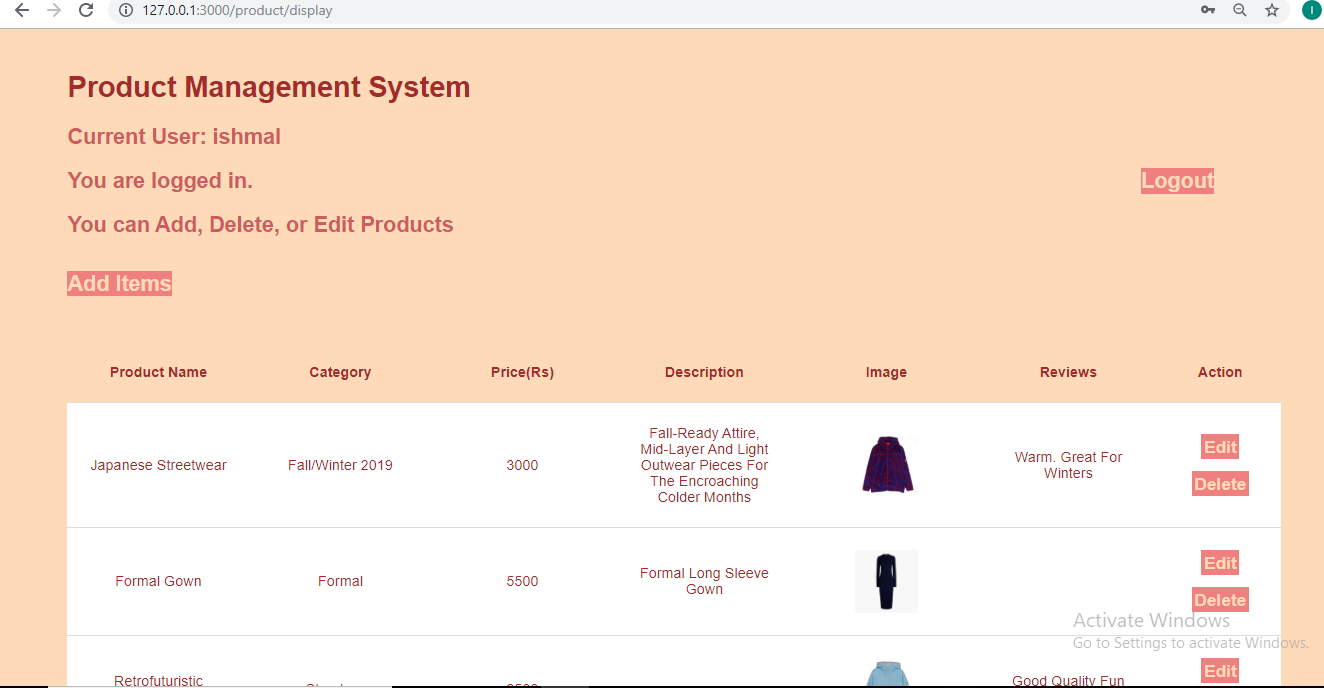
#### If the input fields are incomplete, the signup is not completed and the system prompts the user to enter all the details.



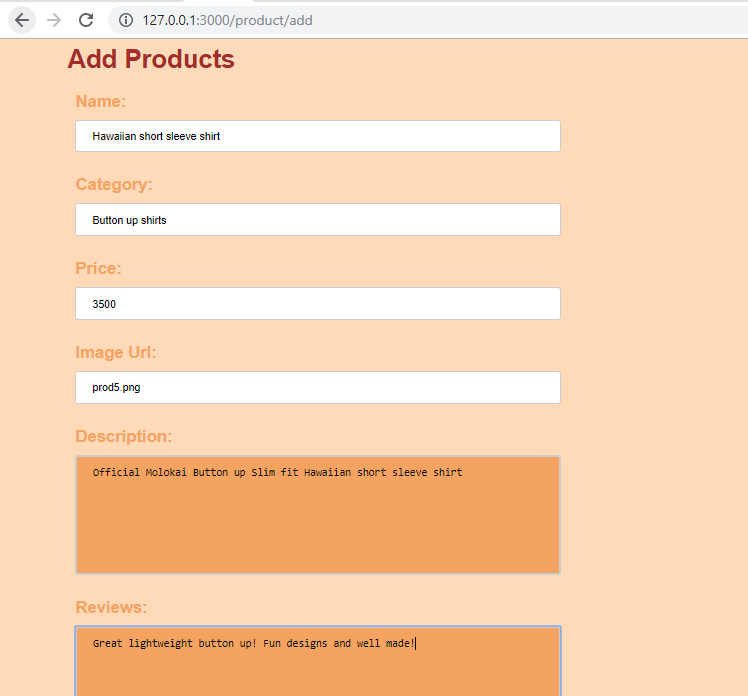
#### Enter valid relevant signup data.

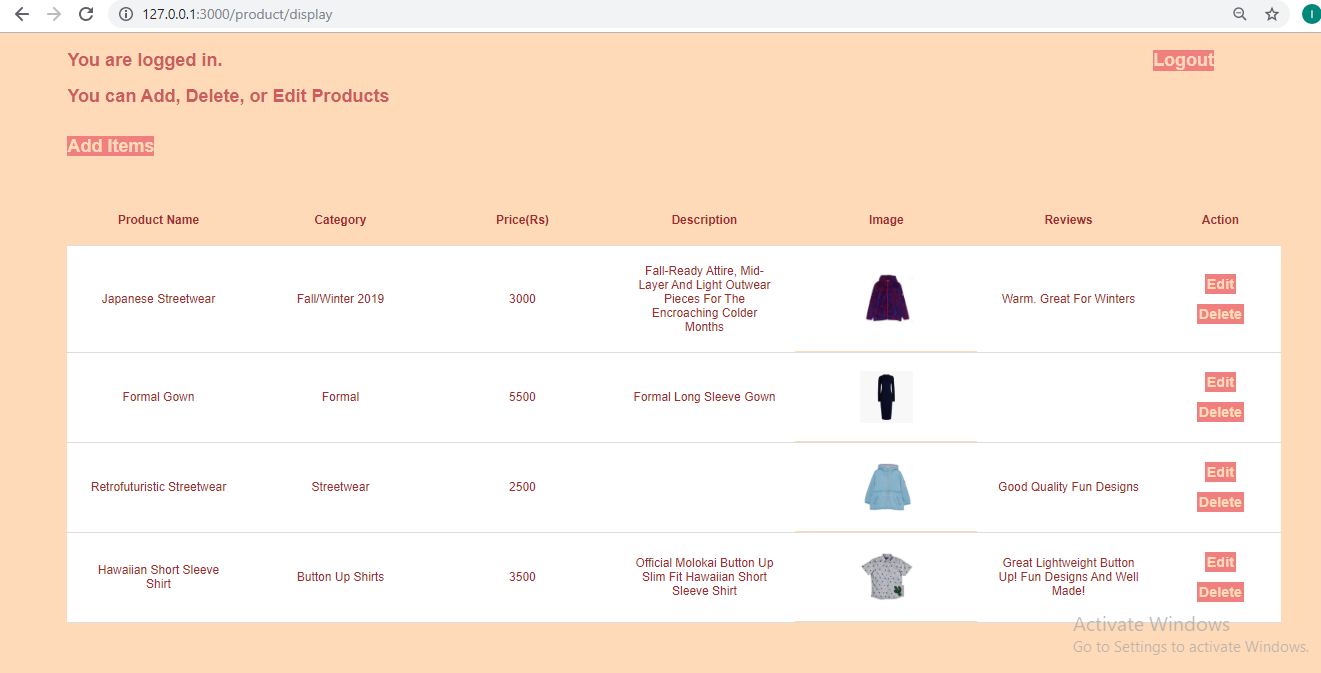


#### When you sign in as a user, you’re able to see all the existing products, edit or delete them, and add new products.



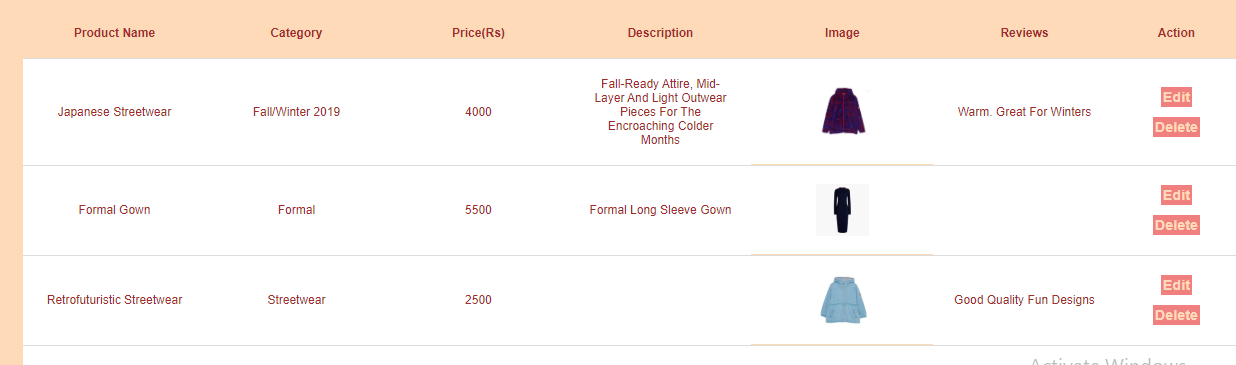
#### Adding a new item:



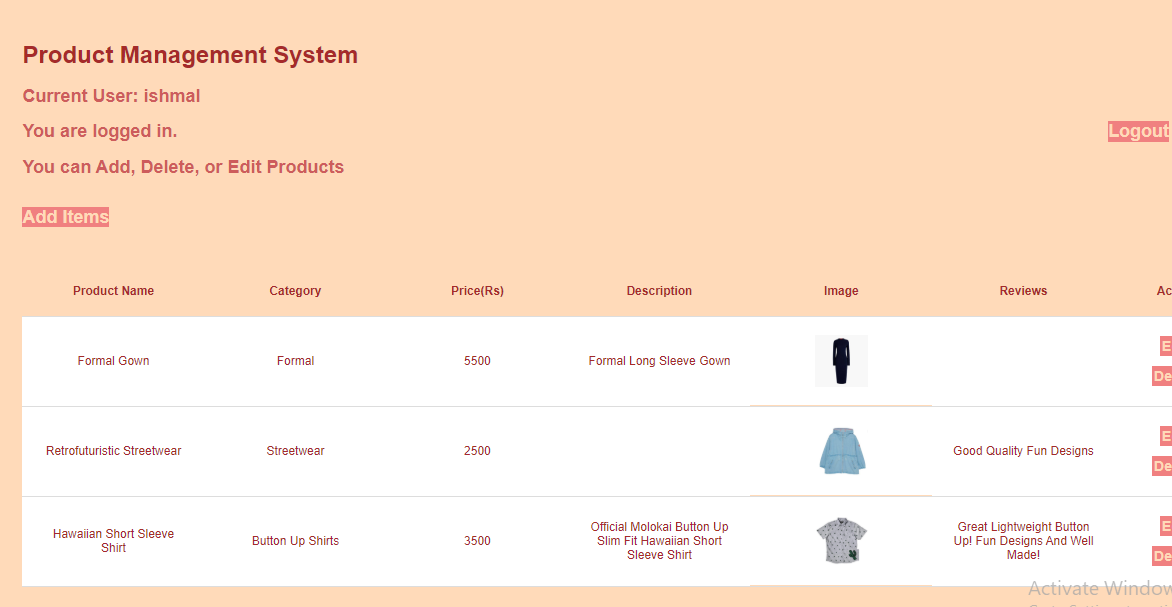


#### Edit the information of any existing product.

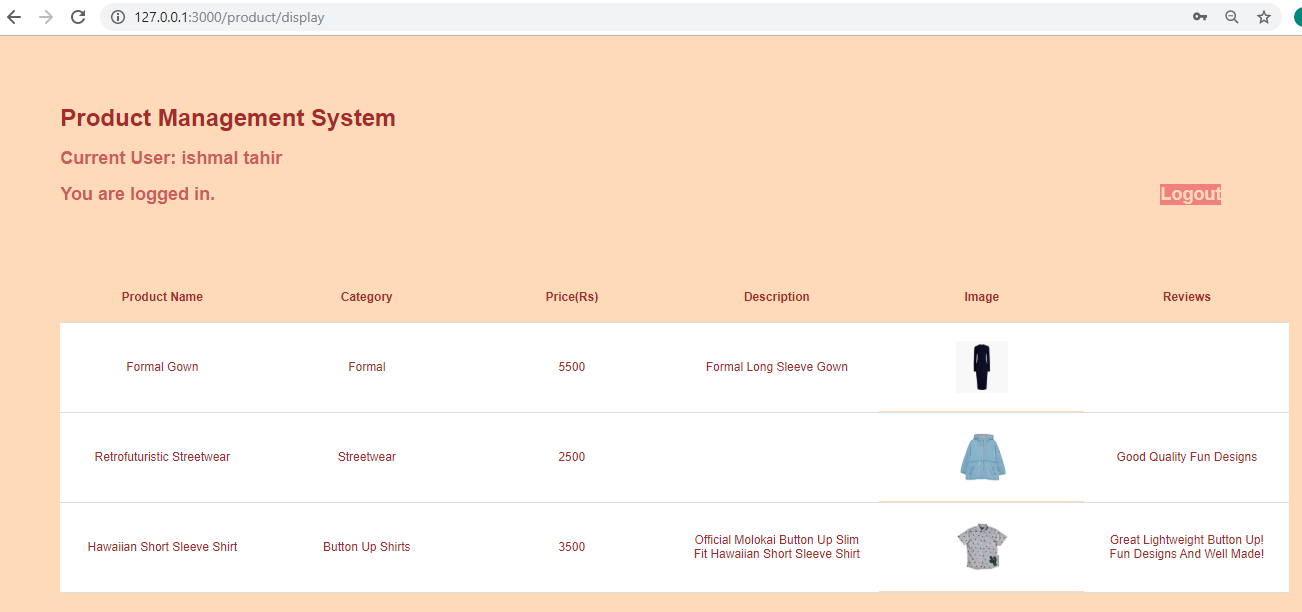




#### Delete a product from selecting the action on right.



#### Log in as a buyer to see the details of existing products



#### Click logout to navigate back to the login page and end your session