**Sri Lanka Institute of Information Technology**

****

**Distributed Systems**

**Assignment 2 – REST API**

**2022S1\_REG\_64**

|  |  |
| --- | --- |
| **Student ID** | **Name with initials** |
| **IT19018560** | **Widanapathirana W.P.V.K** |
| **IT20115920** | **Warnakulasuriya I.S.G** |
| **IT20115852** | **Diwyanjalee K.V.S** |
| **IT20097042** | **Perera J.S** |

Contents

# High level architectural diagram

# Functionality

Code

User

User route

import express from "express";  
import UsersController from "./users.controller.js";  
  
const router = express.*Router*();  
  
router.route("/").post(UsersController.*apiRegister*);  
router.route("/login").post(UsersController.*apiLogin*);  
  
export default router;

User controller

import UsersDAO from "../dao/UsersDAO.js";  
  
export default class UsersController {  
 static async *apiRegister*(req, res, next){  
 const {status, user} = await UsersDAO.*register*(req);  
  
 let response = {  
 status: status,  
 user : user  
 }  
  
 res.send(JSON.stringify(response));  
 }  
  
 static async *apiLogin*(req, res, next){  
 const {status, token, user} = await UsersDAO.*login*(req);  
  
 let response = {  
 status: status,  
 token,  
 user : user  
 }  
  
 res.send(JSON.stringify(response));  
 }  
}

Tickets

Tickets route

import express from "express";  
import TicketsController from "./tickets.controller.js";  
import qr from 'qr-image'  
  
const router = express.*Router*();  
  
router.route("/:id").get(TicketsController.*apiGetTickets*);  
  
router.route("/qr/:code").get((req,res) => {  
 var code = qr.image(req.params.code, { type: 'png' });  
 res.setHeader('Content-type', 'image/png'); *//sent qr image to client side* code.pipe(res);  
})  
  
export default router;

Tickets controller

import TicketsDAO from "../dao/ticketsDAO.js";  
  
  
export default class TicketsController{  
 static async *apiGetTickets*(req, res, next) {  
 const {status, tickets} = await TicketsDAO.*getTickets*(req.params.id);  
  
 let response = {  
 status: status,  
 tickets: tickets,  
 };  
  
 res.json(response);  
 }  
}

Theater

Theater route

import express from "express";  
import banner from "../storage.js";  
import TheatersController from "./theater.controller.js";  
  
const router = express.*Router*();  
  
router.route("/").get(TheatersController.*apiGetTheaters*);  
router.route("/").post(TheatersController.*apiAddTheaters*);  
router.route("/movie/:id").post(TheatersController.*apiaddMoviesToTheater*);  
router.route("/movies/:id").get(TheatersController.*apiGetMovies*);  
router.route("/edit-movie/:id").post(TheatersController.*apiEditMovie*);  
  
router.route("/banner/:title").post(banner.single('image'),TheatersController.*addMovieBanner*);  
  
export default router;

Theater controller

import TheaterDAO from "../dao/theaterDAO.js";  
  
export default class TheatersController {  
 static async *apiGetTheaters*(req, res, next) {  
 const {status, theaters} = await TheaterDAO.*getTheaters*(req);  
  
 let response = {  
 status: status,  
 theaters: theaters  
 };  
  
 res.json(response);  
 }  
  
 static async *apiAddTheaters*(req, res, next) {  
 const {status, theaters} = await TheaterDAO.*addTheaters*(req);  
  
 let response = {  
 status: status,  
 theaters: theaters  
 };  
  
 res.json(response);  
 }  
  
 static async *apiaddMoviesToTheater*(req, res, next) {  
 const {movies} = await TheaterDAO.addMoviesToTheater(req.params.id, req);  
  
 let response = {  
 movies: movies  
 };  
  
 res.json(response);  
 }  
   
 static async *addMovieBanner*(req, res, next){  
 const image = req.image;  
 res.send(JSON.stringify({message: 'File uploaded successfully.', image}));  
 }  
  
 static async *apiGetMovies*(req, res, next){  
 const {status, movies} = await TheaterDAO.*getMovies*(req.params.id);  
  
 let response = {  
 status: status,  
 movies: movies  
 };  
  
 res.json(response);  
 }  
  
 static async *apiEditMovie*(req, res, next){  
 const {status, movies} = await TheaterDAO.editMovies(req.params.id, req);  
  
 let response = {  
 status: status,  
 movies: movies  
 };  
  
 res.json(response);  
 }  
}

Movies

Movies route

import express from "express";  
import MoviesController from "./movies.controller.js";  
  
const router = express.*Router*();  
  
router.route("/").get(MoviesController.*apiGetMovies*);  
router.route("/:id").get(MoviesController.*apiGetMovie*);  
router.route("/").post(MoviesController.*apiAddMovie*);  
router.route("/:id").put(MoviesController.*apiUpdateMovie*);  
router.route("/:id").delete(MoviesController.*apiDeleteMovie*);  
  
  
export default router;

Movies controller

import MoviesDAO from "../dao/moviesDAO.js"  
  
  
export default class MoviesController {  
 static async *apiGetMovies*(req, res, next) {  
 const {movies} = await MoviesDAO.*getMovies*();  
   
 let response = {  
 movies: movies,  
 };  
  
 res.json(response);  
 }  
  
 static async *apiGetMovie*(req, res, next) {  
 const {status, movie} = await MoviesDAO.*getMovie*(req.params.id);  
   
 let response = {  
 status: status,  
 movie: movie,  
 };  
  
 res.json(response);  
 }  
  
 static async *apiAddMovie*(req, res, next){  
 const {status, movie} = await MoviesDAO.*addMovie*(req);  
  
 let response = {  
 status: status,  
 movie : movie,  
 }  
  
 res.json(response);  
 }  
  
 static async *apiUpdateMovie*(req, res, next){  
 const {status, movie} = await MoviesDAO.*updateMovie*(req.params.id, req);  
  
 let response = {  
 status: status,  
 movie: movie,  
 };  
  
 res.json(response);  
 }  
  
 static async *apiDeleteMovie*(req, res, next){  
 const {status, movie} = await MoviesDAO.*deleteMovie*(req.params.id, req);  
  
 let response = {  
 status: status,  
 movie: movie,  
 };  
  
 res.json(response);  
 }  
}

Cart

Cart route

import express from "express";  
import CartController from "./cart.constroller.js";  
  
const router = express.*Router*();  
  
router.route("/").post(CartController.*apiAddToCart*);  
router.route("/:id").get(CartController.*apiGetCart*);  
  
export default router;

Cart controller

import CartDAO from "../dao/cartDAO.js";  
  
export default class CartController{  
 static async *apiAddToCart*(req, res, next) {  
 const {status, movie} = await CartDAO.*addToCart*(req);  
  
 let response = {  
 status: status,  
 movie: movie,  
 };  
  
 res.json(response);  
 }  
  
 static async *apiGetCart*(req, res, next) {  
 const {status, movies} = await CartDAO.*getCart*(req.params.id);  
  
 let response = {  
 status: status,  
 movie: movies,  
 };  
  
 res.json(response);  
 }  
}

Payment

Payment route

import express from "express";  
import PaymentController from "./payments.controller.js";  
  
const router = express.*Router*();  
  
router.route("/").post(PaymentController.*apiAddPayment*);  
  
export default router;

Payment controller

import PaymentsDAO from "../dao/paymentsDAO.js";  
  
export default class PaymentController{  
 static async *apiAddPayment*(req, res, next) {  
 const {status, payment} = await PaymentsDAO.*addPayment*(req);  
  
 let response = {  
 status: status,  
 payment: payment,  
 };  
  
 res.json(response);  
 }  
}

Database

Cart

import mongodb from 'mongodb'  
  
const ObjectId = mongodb.ObjectId  
let cart  
  
export default class CartDAO {  
 static async *injectDB*(conn) {  
 if(cart){  
 return  
 }  
 try {  
 cart = await conn.db(process.env.THEATER\_NS).collection("cart");  
 } catch (e) {  
 console.error(  
 `Unable to establish a colection handle in MoviesDAO: ${e}`,  
 )  
 }  
 }  
  
 static async *addToCart*(req){  
 let cursor;  
  
 try {  
 let obj = {  
 user\_id: req.body.user\_id,  
 movie: req.body.movie,  
 };  
 cursor = await cart.insertOne(obj);  
 return {status: true, movie : cursor};  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, movie : '' };  
 }  
 }  
  
 static async *getCart*(id) {  
 let cursor ;  
  
 try {  
 cursor = await cart.find({user\_id: id}).toArray();  
 return { status:true, movies: cursor };  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status:true, movies : []};  
 }  
 }  
  
 static async *deleteUserCart*(id) {  
 let cursor;  
  
 try {  
 cursor = await cart.deleteMany({user\_id: id})  
 return { status:true, cart: cursor };  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status:true, cart : []};  
 }  
 }  
}

Movies

import mongodb from 'mongodb'  
import TheaterDAO from './theaterDAO.js';  
import url from '../url.js';  
import UsersDAO from './UsersDAO.js';  
  
const ObjectId = mongodb.ObjectId  
let movies  
  
export default class MoviesDAO {  
 static async *injectDB*(conn) {  
 if(movies){  
 return  
 }  
 try {  
 movies = await conn.db(process.env.THEATER\_NS).collection("movies");  
 } catch (e) {  
 console.error(  
 `Unable to establish a colection handle in MoviesDAO: ${e}`,  
 )  
 }  
 }  
  
 static async *getMovies*() {  
 let cursor ;  
  
 try {  
 cursor = await movies.find().toArray();  
 return { movies: cursor };  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { movies : []};  
 }   
 }  
  
 static async *getMovie*(id) {  
 let cursor ;  
  
 try {  
 cursor = await movies.findOne({\_id: ObjectId(id)});  
 return { status: true, movie : cursor };  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, movie : []};  
 }   
 }  
  
 static async *addMovie*(req){  
 let cursor;  
  
 let userAuth = await UsersDAO.*userAuth*(req);  
  
 if(userAuth.status){  
 if(userAuth.user.role === 'movie\_admin'){  
 try {  
 let obj = {  
 title: req.body.title,  
 time : req.body.time,  
 description : req.body.description,  
 cast: req.body.cast,  
 price: req.body.price,  
 banner: url + req.body.banner  
 };  
 cursor = await movies.insertOne(obj);  
 await TheaterDAO.*addMovieToTheater*(req.body.theater\_id, cursor.insertedId);  
 return {status: true, movie : cursor};  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, movie : '' };  
 }  
 }  
 }  
 return { status: false, movie : 'Unauthorized User' };  
   
 }  
  
 static async *updateMovie*(id ,req){  
 let cursor;  
  
 let userAuth = await UsersDAO.*userAuth*(req);  
  
 if(userAuth.status){  
 if(userAuth.user.role === 'movie\_admin'){  
 try {  
 let obj = {  
 title: req.body.title,  
 description: req.body.description,  
 cast: req.body.cast,  
 time: req.body.time,  
 price: req.body.price,  
 };  
   
 cursor = await movies.updateOne({\_id: ObjectId(id)}, {$set:obj});  
 return {status: true, movie : cursor };  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, movie : '' };  
 }  
 }  
 }  
  
 return { status: false, movie : 'Unauthorized User' };  
 }  
  
 static async *deleteMovie*(id, req){  
 let cursor;  
  
 let userAuth = await UsersDAO.*userAuth*(req);  
  
 if(userAuth.status){  
 if(userAuth.user.role === 'movie\_admin'){  
 try {  
 cursor = await movies.deleteOne({\_id: ObjectId(id)});  
 return { status: true, movie : cursor};  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status:false, movie : '' };  
 }  
 }  
 }  
   
 return { status: false, movie : 'Unauthorized User' };  
 }  
}

Payment

import mongodb from 'mongodb'  
import CartDAO from './cartDAO.js';  
import TicketsDAO from './ticketsDAO.js';  
import nodemailer from 'nodemailer'  
  
const ObjectId = mongodb.ObjectId  
let payments  
  
export default class PaymentsDAO {  
 static async *injectDB*(conn) {  
 if(payments){  
 return  
 }  
 try {  
 payments = await conn.db(process.env.THEATER\_NS).collection("payments");  
 } catch (e) {  
 console.error(  
 `Unable to establish a colection handle in MoviesDAO: ${e}`,  
 )  
 }  
 }  
  
 static async *addPayment*(req){  
 let cursor;  
  
 try {  
 let obj = {  
 user\_id: req.body.user\_id,  
 total: req.body.total,  
 };  
 cursor = await payments.insertOne(obj);  
 await CartDAO.*deleteUserCart*(req.body.user\_id);  
 await TicketsDAO.*addTicket*(req);  
  
 var transporter = nodemailer.createTransport({  
 service: 'gmail',  
 auth: {  
 user: 'ninoauthenticator@gmail.com',  
 pass: 'Nino@123'  
 }  
 });  
   
 var mailOptions = {  
 from: 'ninoauthenticator@gmail.com',  
 to: 'indurusw@gmail.com',  
 subject: 'Payment',  
 text: 'payment is successful!'  
 };  
   
 transporter.sendMail(mailOptions, function(error, info){  
 if (error) {  
 console.log(error);  
 } else {  
 console.log('Email sent: ' + info.response);  
 }  
 });  
  
 return {status: true, payment : cursor};  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, payment : '' };  
 }  
 }  
}

theater

import mongodb from 'mongodb';  
import url from '../url.js';  
import {*randomBytes*} from 'crypto';  
import UsersDAO from './UsersDAO.js';  
  
const ObjectId = mongodb.ObjectId;  
let theaters;  
  
export default class TheaterDAO {  
 static async *injectDB*(conn) {  
 if(theaters){  
 return  
 }  
 try{  
 theaters = await conn.db(process.env.THEATER\_NS).collection("theaters");  
 } catch (e) {  
 console.error(  
 `Unable to establish a colection handle in MoviesDAO: ${e}`,  
 )  
 }  
 }  
  
 static async *getTheaters*(req) {  
 let cursor ;  
  
 let userAuth = await UsersDAO.*userAuth*(req);  
  
 if(userAuth.status){  
 try {  
 cursor = await theaters.find().toArray();  
 return { status: true, theaters: cursor };  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, theaters : [] };  
 }  
 }  
  
 return { status: false, theaters : 'unauthorized user' };  
 }  
  
 static async *addTheaters*(req){  
 let cursor;  
  
 let userAuth = await UsersDAO.*userAuth*(req);  
  
 if(userAuth.status){  
 if(userAuth.user.role === 'system\_admin'){  
 try {  
 let obj = {  
 name : req.body.name,  
 };  
 cursor = await theaters.insertOne(obj);  
 return { status: true, theater : cursor};  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, theater : '' };  
 }  
 }  
 }  
  
 return { status: false, theater: 'unauthorized user' };  
 }  
  
 *// static async addMoviesToTheater(id, req){  
 // let cursor;  
  
 // try {  
 // let obj = {   
 // movies : {  
 // id: randomBytes(10).toString('hex'),  
 // title: req.body.title,  
 // time : req.body.time,  
 // description : req.body.description,  
 // cast: req.body.cast,  
 // price: req.body.price,  
 // banner: url + req.body.banner  
 // }  
 // };  
  
 // cursor = await theaters.updateOne({\_id: ObjectId(id)}, {$push:obj});  
 // return {movie : cursor};  
 // } catch (e) {  
 // console.error(`Unable to issues find command, ${e}`);  
 // return { movie : '' };  
 // }  
 // }* static async *addMovieToTheater*(id, movie){  
 let cursor;  
  
 let obj = {  
 movies: movie  
 }  
 cursor = await theaters.updateOne({\_id: ObjectId(id)}, {$push:obj});  
 return {movie : cursor};  
 }  
  
 static async *getMovies*(id){  
 let cursor;  
  
 try{  
 cursor = await theaters.findOne({\_id: ObjectId(id)});  
 let movies = cursor.movies;  
 return { status: true, movies: movies };  
 }catch(e){  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, movies : '' };  
 }  
 }  
}

Tickets

import mongodb from 'mongodb'  
import {*randomBytes*} from 'crypto';  
import qr from 'qr-image'  
  
const ObjectId = mongodb.ObjectId  
let tickets  
  
export default class TicketsDAO {  
 static async *injectDB*(conn) {  
 if(tickets){  
 return  
 }  
 try {  
 tickets = await conn.db(process.env.THEATER\_NS).collection("tickets");  
 } catch (e) {  
 console.error(  
 `Unable to establish a colection handle in MoviesDAO: ${e}`,  
 )  
 }  
 }  
  
 static async *addTicket*(req){  
 let cursor;  
  
 try {  
 req.body.movies.map(async(movie) => {  
 let qr\_code = *randomBytes*(15).toString('hex');  
 let obj = {  
 user\_id: req.body.user\_id,  
 qr\_code: qr\_code,  
 movie: movie.movie  
 };  
 cursor = await tickets.insertOne(obj);  
 });  
   
 return {status: true, ticket : cursor};  
 } catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, ticket : '' };  
 }  
 }  
  
 static async *getTickets*(id){  
 let cursor;  
  
 try {  
 cursor = await tickets.find({user\_id:id}).toArray();  
 return {status: true, tickets: cursor}  
 }  
 catch(e){  
 console.error(`Unable to issues find command, ${e}`);  
 return { status: false, tickets : '' };  
 }  
 }  
}

users

import mongodb from 'mongodb';  
import {*randomBytes*} from 'crypto';  
import *hashedPassword* from '../hashedPassword.js';  
  
const ObjectId = mongodb.ObjectId;  
let users;  
  
export default class UsersDAO{  
 static async *injectDB*(conn){  
 if(users){  
 return;  
 }  
 try{  
 users = await conn.db(process.env.THEATER\_NS).collection("users");  
 } catch (e) {  
 console.error(`Unable to establish a colection handle in MoviesDAO: ${e}`);  
 }   
 }  
  
 static async *userAuth*(req){  
 try {  
 let user = await users.findOne({token: req.headers.authorization});  
  
 if(user != null){  
 return {status: true, user : user};  
 }  
  
 return {status: false, user: null};  
 }  
 catch (e) {  
 console.error(`Unable to issues find command, ${e}`);  
 return {status: false, user : null};  
 }  
 }  
  
 static async *register*(req) {  
 try {  
 let user = await users.findOne({email: req.body.email});  
  
 if(user == null){  
 var obj = {  
 email: req.body.email,  
 password: *hashedPassword*(req.body.password),  
 role: req.body.role,  
 }  
   
 await users.insertOne(obj);  
 return {status: true, user : "User Created"};  
 }  
  
 return {status: false, user : "Email is in use"};  
 }  
 catch(e){  
 console.error(`Unable to issues find command, ${e}`);  
 return {status: false, user : ""};  
 }  
 }  
  
 static async *login*(req){  
 try{  
 let user = await users.findOne({email: req.body.email});  
 if(user != null){  
 if(user.email === req.body.email && user.password === *hashedPassword*(req.body.password)){  
 let authToken = *randomBytes*(30).toString('hex');  
 let token = {token : authToken}  
 await users.updateOne({\_id: ObjectId(user.\_id)}, {$push:token});  
 return {status:true, token: authToken, user: user};  
 }  
   
 return {status:true, token: null, user : "invalid credential"};  
 }  
  
 return {status:true, token: null, user : "user not found"};  
 }  
 catch(e){  
 console.error(`Unable to issues find command, ${e}`);  
 return {user : ""};  
 }  
 }  
}

server.js

import express from "express";  
import *cors* from 'cors';  
import *bodyParser* from "body-parser";  
  
import theater from "./api/theater.route.js";  
import users from "./api/users.route.js";  
import cart from "./api/cart.route.js";  
import payments from "./api/payments.route.js";  
import tickets from "./api/tickets.route.js";  
import movies from "./api/movies.route.js";  
  
const app = express();  
  
app.use(*cors*());  
app.use(bodyParser.json());  
app.use(express.static('public'));  
app.use('/banners', express.static('/banners'));  
  
app.use("/api/users", users);  
app.use("/api/theaters", theater);  
app.use("/api/cart", cart);  
app.use("/api/payments", payments);  
app.use("/api/tickets", tickets);  
app.use("/api/movies", movies);  
app.use("\*", (req, res) => res.status(404).json({error : "not found"}));  
  
export default app;

Frontend

app.jsx

import React from "react";  
import {*createRoot*} from "react-dom/client";  
import {BrowserRouter, Routes, Route} from "react-router-dom";  
import 'bootstrap/dist/css/bootstrap.min.css';  
  
import *Main* from "./components/Main";  
import *Register* from "./components/Register";  
import *Theaters* from "./components/system\_admin/Theaters";  
import *Movies* from "./components/system\_admin/Movies";  
import *AddTheater* from "./components/system\_admin/AddTheaters";  
import *MoviesAdmin* from "./components/movie\_admin/MoviesAdmin";  
import *AddMovies* from "./components/movie\_admin/AddMovies";  
import *Home* from "./components/customer/Home";  
import *Cart* from "./components/customer/Cart";  
import *Payments* from "./components/customer/payment/Payments";  
import *Tickets* from "./components/customer/Tickets";  
import *EditMovie* from "./components/movie\_admin/EditMovie";  
  
*createRoot*(document.getElementById('app')).render(<App/>);  
  
function *App*(){  
 return (  
 <BrowserRouter>  
 <Routes>  
 <Route path="/" element={<Main/>}/>  
 <Route path="/register" element={<Register/>}/>  
 <Route path="/theaters" element={<Theaters/>}/>  
 <Route path="/add-theaters" element={<AddTheater/>}/>  
 <Route path="/movies" element={<Movies/>}/>  
 <Route path="/movie-admin" element={<MoviesAdmin/>}/>  
 <Route path="/add-movies" element={<AddMovies/>}/>  
 <Route path="/edit-movie" element={<EditMovie/>}/>  
 <Route path="/home" element={<Home/>}/>  
 <Route path="/cart" element={<Cart/>}/>  
 <Route path="/tickets" element={<Tickets/>}/>  
 <Route path="/payments" element={<Payments/>}/>  
 </Routes>  
 </BrowserRouter>  
 )  
}  
  
export default *App*;

Main.jsx

import React, {*useEffect*, *useState*} from "react";  
import { Form, Button } from "react-bootstrap";  
import { *Link*, useNavigate } from "react-router-dom";  
  
export default function *Main*(){  
 const [email, setEmail] = *useState*("");  
 const [password, setPassword] = *useState*("");  
  
 const navigate = useNavigate();  
  
 *useEffect*(()=> {  
 if(localStorage.getItem("role") === "customer"){  
 navigate('/home');  
 }  
 else if(localStorage.getItem("role") === "movie\_admin"){  
 navigate('/movie-admin');  
 }  
 else if(localStorage.getItem("role") === "system\_admin"){  
 navigate('/theaters');  
 }  
 });  
  
 async function login(e){  
 e.preventDefault();  
   
 await *fetch*("http://localhost:5000/api/users/login", {  
 method: "POST",  
 headers: {'Content-Type': 'application/json'},  
 body: JSON.stringify({  
 email: email,  
 password: password,  
 }  
 )  
 })  
 .then(response => response.json())  
 .then(user => {  
 localStorage.setItem('token', user.token);  
 localStorage.setItem('role', user.user.role);  
 localStorage.setItem('user\_id', user.user.\_id);  
  
 if(user.token != null){  
 if(user.user.role === "customer"){  
 navigate('/home');  
 }  
 else if(user.user.role === "movie\_admin"){  
 navigate('/movie-admin');  
 }  
 else if(user.user.role === "system\_admin"){  
 navigate('/theaters');  
 }  
 }  
 });  
   
 }  
  
 return <Form style={{margin: 100}} onSubmit={login}>  
 <h1>Login</h1>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Username</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setEmail(e.target.value)}  
 type="text"  
 value={email}  
 placeholder="Username"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formBasicPassword">  
 <Form.Label>Password</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setPassword(e.target.value)}  
 value={password}  
 type="password"  
 placeholder="Password"  
 />  
 </Form.Group>  
 <span>Don't have an account? </span>  
 <Link to={'/register'}>register</Link>  
 <br/>  
 <Button variant="primary" type="submit">  
 Submit  
 </Button>  
 </Form>  
  
}

Register.jsx

import React, {*useState*} from "react";  
import { Form, Button } from "react-bootstrap";  
import { *Link*, useNavigate } from "react-router-dom";  
  
export default function *Register*(){  
 const [email, setEmail] = *useState*("");  
 const [role, setRole] = *useState*("customer");  
 const [password, setPassword] = *useState*("");  
  
 const navigate = useNavigate();  
  
 async function register(e){  
 localStorage.clear();  
 e.preventDefault();  
   
 await *fetch*("http://localhost:5000/api/users/", {  
 method: "POST",  
 headers: {'Content-Type': 'application/json'},  
 body: JSON.stringify({  
 email: email,  
 password: password,  
 role: role,  
 }  
 )  
 })  
 .then(response => response.json())  
 .then(user => {  
 if(user.status){  
 *alert*("User created");  
 navigate('/');  
 }  
 else{  
 *alert*(user.user);  
 }  
   
 });  
   
 }  
  
 return <Form style={{margin: 100}} onSubmit={register}>  
 <h1>Register</h1>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Email</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setEmail(e.target.value)}  
 type="text"  
 value={email}  
 placeholder="Email"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3">  
 <Form.Label>Role</Form.Label>  
 <Form.Select onChange={(e) => setRole(e.target.value)} value={role}>  
 <option value="customer">Customer</option>  
 <option value="movie\_admin">Movie Admin</option>  
 </Form.Select>  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formBasicPassword">  
 <Form.Label>Password</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setPassword(e.target.value)}  
 value={password}  
 type="password"  
 placeholder="Password"  
 />  
 </Form.Group>  
 <span>Already have an account? </span>  
 <Link to={'/'}>login</Link>  
 <br/>  
 <Button variant="primary" type="submit">  
 Submit  
 </Button>  
 </Form>  
  
}

System Admin

AddTheaters.jsx

import React, {*useEffect*, *useState*} from "react";  
import { Form, Button } from "react-bootstrap";  
import { *Link*, useNavigate } from "react-router-dom";  
import *SystemAdminNavbar* from "../navs/SystemAdminNavbar";  
import *TheaterCard* from "./TheaterCard";  
  
export default function *AddTheater*(){  
 const [name, setname] = *useState*("");  
  
 const navigate = useNavigate();  
  
 async function addTheater(e){  
 e.preventDefault();  
   
 await *fetch*("http://localhost:5000/api/theaters", {  
 method: "POST",  
 headers: {  
 'Content-Type': 'application/json',  
 'Authorization' : localStorage.getItem('token'),  
 },  
 body: JSON.stringify({  
 name: name,  
 }  
 )  
 })  
 .then(response => response.json())  
 .then(user => {  
 if(!user.status){  
 *alert*('Unautorized User');  
 navigate('/');  
 }  
 else{  
 *alert*('Theater Added');  
 }  
 });  
   
 }  
  
 return <>  
 <SystemAdminNavbar/>  
 <Form style={{margin: 100}} onSubmit={addTheater}>  
 <h1>Add New Theater</h1>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Theater Name</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setname(e.target.value)}  
 type="text"  
 value={name}  
 placeholder="Theater name"  
 />  
 </Form.Group>  
 <Button variant="primary" type="submit">  
 Submit  
 </Button>  
 </Form>  
 </>  
}

MovieCard.jsx

import { *useEffect*, *useState* } from "react";  
import { Card, Button } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
  
export default function *MovieCard*(props) {  
 const {id} = props;  
  
 const [movie, setMovie] = *useState*();  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/movies/"+id)  
 .then(response => response.json())  
 .then(item => {  
 setMovie(item.movie);  
 });  
 }  
 fetchItems();  
   
 }, []);  
  
 if(movie){  
 return <Card style={{ width: '18rem' }}>  
 <Card.Img variant="top" src={movie.banner} />  
 <Card.Body style={{textAlign:"center"}}>  
 <Card.Title>{movie.title}</Card.Title>  
 <Card.Text>  
 {movie.description}  
 </Card.Text>  
 <Card.Text>  
 {movie.cast}  
 </Card.Text>  
 <Card.Text>  
 <b>Time:</b> {movie.time}  
 </Card.Text>  
 <Card.Text>  
 <b>Ticket Price:</b> Rs.{movie.price}  
 </Card.Text>  
 </Card.Body>  
 </Card>  
 }  
  
   
}

TheaterCard.jsx

import { Card, Button } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
  
export default function *TheaterCard*(props) {  
 const {theater} = props;  
  
  
 const navigate = useNavigate();  
  
 async function navigateToMovies(){  
 navigate('/movies/?id='+theater.\_id);  
 }  
  
 return <Card style={{width:'10rem' ,margin: 20}}>  
 <Card.Body>  
 <Card.Title>  
 {theater.name}  
 </Card.Title>  
 <Button variant="primary" onClick={navigateToMovies}>See Movies</Button>  
 </Card.Body>  
 </Card>  
}

Movies.jsx

import { *useEffect*, *useState* } from "react";  
import { Card, Button } from "react-bootstrap";  
import { useSearchParams } from "react-router-dom";  
import *SystemAdminNavbar* from "../navs/SystemAdminNavbar";  
import *MovieCard* from "./MovieCard";  
  
export default function Movies() {  
 const [searchParams, setSearchParams] = useSearchParams();  
 const [movies, setMovies] = *useState*([]);  
  
 const id = searchParams.get('id');  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/theaters/movies/"+id)  
 .then(response => response.json())  
 .then(items => {  
 setMovies(items.movies);  
 });  
 }  
 fetchItems();  
 }, []);  
  
 if(movies){  
 return <>  
 <SystemAdminNavbar/>  
 <div style={{margin: 30, display:"flex", flexWrap:"wrap"}}>  
 { movies.map(movie => <MovieCard key={movie.\_id} id={movie}/>) }  
 </div>  
 </>  
 }  
 else {  
 return<>  
 <SystemAdminNavbar/>  
 <h1>Movies Not Found!</h1>  
 </>  
   
 }  
  
   
}

Theaters.jsx

import React, {*useEffect*, *useState*} from "react";  
import *SystemAdminNavbar* from "../navs/SystemAdminNavbar";  
import *TheaterCard* from "./TheaterCard";  
  
export default function *Theaters*(){  
 const [theaters, setTheaters] = *useState*([]);  
 const [auth, setAuth] = *useState*(false);  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/theaters/", {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(items => {  
 if(items.status) {  
 setAuth(true);  
 setTheaters(items.theaters);  
 }  
 })  
 }  
  
 fetchItems();  
 }, []);  
  
   
  
 if(auth){  
 return (  
 <>  
 <SystemAdminNavbar/>  
 {  
 theaters.map(theater =>   
 <TheaterCard key={theater.\_id} theater={theater}/>  
 )  
 }  
 </>  
 )  
 }  
   
 return <h1>Unauthorized User</h1>  
}

Movie Admin

AddMovies.jsx

import React, {*useEffect*, *useState*} from "react";  
import { Form, Button } from "react-bootstrap";  
import { useNavigate, useSearchParams } from "react-router-dom";  
import *MovieAdminNavbar* from "../navs/MovieAdminNavbar";  
  
export default function *AddMovies*(){  
 const [searchParams, setSearchParams] = useSearchParams();  
 const [movies, setMovies] = *useState*([]);  
  
 const [title, setTitle] = *useState*("");  
 const [description, setDescription] = *useState*("");  
 const [cast, setCast] = *useState*("");  
 const [time, setTime] = *useState*("");  
 const [price, setPrice] = *useState*("");  
 const [banner, setBanner] = *useState*();  
  
 const navigate = useNavigate();  
 const id = searchParams.get('id');   
  
 async function addMovie(e){  
 e.preventDefault();  
  
 const formData = new *FormData*();  
 formData.append('image', banner);  
  
 await *fetch*("http://localhost:5000/api/theaters/banner/"+title,{  
 method: "POST",  
 body: formData,  
 });  
   
 await *fetch*("http://localhost:5000/api/movies/", {  
 method: "POST",  
 headers: {  
 'Content-Type': 'application/json',  
 'Authorization' : localStorage.getItem('token'),  
 },  
 body: JSON.stringify({  
 theater\_id: id,  
 title: title,  
 time: time,  
 description: description,  
 cast: cast,  
 price: price,  
 banner: title + '-'+ banner.name  
 }  
 )  
 })  
 .then(() => {  
 *alert*("Movie Added");  
 navigate('/movie-admin');  
 });  
 }  
  
 return <>  
 <MovieAdminNavbar/>  
 <Form style={{margin: 100}} onSubmit={addMovie}>  
 <h1>Add New Movie</h1>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Movie Title</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setTitle(e.target.value)}  
 type="text"  
 value={title}  
 placeholder="Movie Title"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Description</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setDescription(e.target.value)}  
 type="text"  
 value={description}  
 placeholder="Description"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Movie Cast</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setCast(e.target.value)}  
 type="text"  
 value={cast}  
 placeholder="Movie Cast"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Show Time</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setTime(e.target.value)}  
 type="time"  
 value={time}  
 placeholder="Show Time"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Ticket Price</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setPrice(e.target.value)}  
 type="number"  
 value={price}  
 placeholder="Price"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Movie Banner</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setBanner(e.target.files[0])}  
 type="file"  
 placeholder="Movie Banner"  
 />  
 </Form.Group>  
 <Button variant="primary" type="submit">  
 Submit  
 </Button>  
 </Form>  
 </>  
}

AdminMovieCard.jsx

import { *useState*, *useEffect* } from "react";  
import { Card, Button } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
  
export default function *AdminMovieCard*(props) {  
 const {id} = props;  
 const [movie, setMovie] = *useState*();  
  
 const navigate = useNavigate();  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/movies/"+id)  
 .then(response => response.json())  
 .then(item => {  
 setMovie(item.movie);  
 });  
 }  
 fetchItems();  
 }, []);  
  
 function navigateToEdit(){  
 navigate('/edit-movie/?id='+id);  
 }  
  
 async function deleteMovie(){  
 await *fetch*("http://localhost:5000/api/movies/"+movie.\_id, {  
 method: "DELETE",  
 headers: {  
 'Content-Type': 'application/json',  
 'Authorization' : localStorage.getItem('token'),  
 },  
 })  
 .then(() => {  
 *alert*("Movie Updated");  
 navigate('/movie-admin');  
 });  
 }  
  
 if(movie){  
 return <div style={{marginTop: 30, display:"flex", flexWrap:"wrap"}}>  
 {  
 <Card style={{ width: '18rem', margin: 5}}>  
 <Card.Img variant="top" src={movie.banner} style={{height:'18rem'}} />  
 <Card.Body style={{textAlign:"center"}}>  
 <Card.Title>{movie.title}</Card.Title>  
 <Card.Text>  
 {movie.description}  
 </Card.Text>  
 <Card.Text>  
 {movie.cast}  
 </Card.Text>  
 <Card.Text>  
 <b>Time:</b> {movie.time}  
 </Card.Text>  
 <Card.Text>  
 <b>Ticket Price:</b> Rs.{movie.price}  
 </Card.Text>  
 <Button style={{margin: 10}} onClick={navigateToEdit}>Edit Movie</Button>  
 <Button style={{margin: 10}} onClick={deleteMovie}>Delete Movie</Button>  
 </Card.Body>  
 </Card>  
 }  
   
 </div>  
  
 }  
  
   
}

EditMovie. Jsx

import React, {*useEffect*, *useState*} from "react";  
import { Form, Button } from "react-bootstrap";  
import { useNavigate, useSearchParams } from "react-router-dom";  
import *MovieAdminNavbar* from "../navs/MovieAdminNavbar";  
  
export default function *EditMovie*(){  
 const [searchParams, setSearchParams] = useSearchParams();  
 const [movie, setMovie] = *useState*();  
  
 const [title, setTitle] = *useState*("");  
 const [description, setDescription] = *useState*("");  
 const [cast, setCast] = *useState*("");  
 const [time, setTime] = *useState*("");  
 const [price, setPrice] = *useState*("");  
  
 const navigate = useNavigate();  
 const id = searchParams.get('id');   
   
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/movies/"+id)  
 .then(response => response.json())  
 .then(item => {  
 setMovie(item.movie);  
 setTitle(item.movie.title);  
 setDescription(item.movie.description);  
 setCast(item.movie.cast);  
 setTime(item.movie.time);  
 setPrice(item.movie.price);  
 });  
 }  
 fetchItems();  
 }, []);  
  
 async function editMovie(e){  
 e.preventDefault();  
   
 await *fetch*("http://localhost:5000/api/movies/"+movie.\_id, {  
 method: "PUT",  
 headers: {  
 'Content-Type': 'application/json',  
 'Authorization' : localStorage.getItem('token'),  
 },  
 body: JSON.stringify({  
 title: title,  
 time: time,  
 description: description,  
 cast: cast,  
 price: price,  
 }  
 )  
 })  
 .then(() => {  
 *alert*("Movie Updated");  
 navigate('/movie-admin');  
 });  
 }  
  
 return <>  
 <MovieAdminNavbar/>  
 <Form style={{margin: 100}} onSubmit={editMovie}>  
 <h1>Edit Movie</h1>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Movie Title</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setTitle(e.target.value)}  
 type="text"  
 value={title}  
 placeholder="Movie Title"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Description</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setDescription(e.target.value)}  
 type="text"  
 value={description}  
 placeholder="Description"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Movie Cast</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setCast(e.target.value)}  
 type="text"  
 value={cast}  
 placeholder="Movie Cast"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Show Time</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setTime(e.target.value)}  
 type="time"  
 value={time}  
 placeholder="Show Time"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Ticket Price</Form.Label>  
 <Form.Control  
 required  
 onChange={(e) => setPrice(e.target.value)}  
 type="number"  
 value={price}  
 placeholder="Price"  
 />  
 </Form.Group>  
 <Button variant="primary" type="submit">  
 Submit  
 </Button>  
 </Form>  
 </>  
}

MovieDetailsCard.jsx

import { *useState*, *useEffect* } from "react";  
import *AdminMovieCard* from "./AdminMovieCard";  
  
export default function *MovieDetailsCard*(props) {  
 const {id} = props;  
 const [movies, setMovies] = *useState*([]);  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/theaters/movies/"+id)  
 .then(response => response.json())  
 .then(items => {  
 setMovies(items.movies);  
 });  
 }  
 fetchItems();  
 }, []);  
   
 if(movies){  
 return <div style={{marginTop: 30, display:"flex", flexWrap:"wrap"}}>  
 {  
 movies.map(movie => <AdminMovieCard id={movie}/>)  
 }  
   
 </div>  
  
 }  
  
   
}

MoviesAdmin.jsx

import { Button } from "react-bootstrap";  
import React, {*useEffect*, *useState*} from "react";  
import *MovieAdminNavbar* from "../navs/MovieAdminNavbar";  
import *TheaterCard* from "../system\_admin/TheaterCard";  
import *TheaterRow* from "./TheaterRow";  
  
export default function *MoviesAdmin*(){  
 const [theaters, setTheaters] = *useState*([]);  
 const [auth, setAuth] = *useState*(false);  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/theaters/", {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(items => {  
 if(items.status) {  
 setAuth(true);  
 setTheaters(items.theaters);  
 }  
 })  
 }  
  
 fetchItems();  
 }, []);  
  
   
  
 if(auth){  
 return (  
 <>  
 <MovieAdminNavbar/>  
 {  
 theaters.map(theater => <>  
 <TheaterRow key={theater.\_id} theater={theater}/>  
 </>  
 )  
 }  
 </>  
 )  
 }  
   
 return <h1>Unauthorized User</h1>  
}

TheaterRow.jsx

import { Card, Button } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
import *MovieDetailsCard* from "./MovieDetailsCard";  
  
export default function *TheaterRow*(props) {  
 const {theater} = props;  
  
 const navigate = useNavigate();  
  
 async function navigateToAddMovies(){  
 navigate('/add-movies/?id='+theater.\_id);  
 }  
  
 return <div style={{margin: 30}}>  
 <h2>{theater.name}</h2>  
 <Button style={{marginLeft: 10}} onClick={navigateToAddMovies}>Add Movie</Button>  
 <MovieDetailsCard id={theater.\_id}/>  
 </div>  
}

Customer

AllMovies.jsx

import { *useEffect*, *useState* } from "react";  
import *HomeMovieCard* from "./HomeMovieCard";  
  
export default function *AllMovies*(props){  
 const {theater} = props;  
 const [movies, setMovies] = *useState*("")  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/theaters/movies/"+ theater.\_id, {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(items => {  
 setMovies(items.movies);  
 })  
 }  
  
 fetchItems();  
 }, []);  
  
 if(movies){  
 return <div style={{marginTop: 30, display:"flex", flexWrap:"wrap"}}>  
 {movies.map(movie => <HomeMovieCard key={movie.time} id={movie}/>)}  
 </div>  
 }  
  
   
   
   
}

Cart.jsx

import { *useEffect*, *useState* } from "react";  
import { Table, Button } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
import *CustomerNavbar* from "../navs/CustomerNavbar";  
  
export default function *Cart*(){  
 const [items, setItems] = *useState*([]);  
 const [total, setTotal] = *useState*(0);  
  
 const navigate = useNavigate();  
  
 function navigateToPayment(){  
 navigate('/payments');  
 }  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/cart/"+localStorage.getItem('user\_id'), {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(item => {  
 let \_total = 0;  
 setItems(item.movie);  
 item.movie.map(movie => {  
 \_total = \_total + *parseInt*(movie.movie.price);  
 setTotal(\_total);  
 })  
 })  
 }  
  
 fetchItems();  
 }, []);  
  
 return <>  
 <CustomerNavbar/>  
 <div style={{margin: 30}}>  
 <Table striped bordered hover size="sm" style={{width:400}}>  
 <thead>  
 <tr>  
 <th>Movie</th>  
 <th>Price</th>  
 </tr>  
 </thead>  
 <tbody>   
 {  
 items.map(item => <tr>  
 <td>{item.movie.title}</td>  
 <td>Rs.{item.movie.price}</td>  
 </tr>)  
 }  
 </tbody>  
 </Table>  
 <h1>Total : Rs.{total}</h1>  
 <Button onClick={navigateToPayment}>Checkout</Button>  
 </div>  
 </>  
}

Home.jsx

import { ModalTitle } from "react-bootstrap";  
import React, {*useEffect*, *useState*} from "react";  
import *CustomerNavbar* from "../navs/CustomerNavbar";  
import *AllMovies* from "./AllMovies";  
  
export default function *Home*(){  
 const [theaters, setTheaters] = *useState*([]);  
 const [auth, setAuth] = *useState*(false);  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/theaters/", {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(items => {  
 if(items.status) {  
 setAuth(true);  
 setTheaters(items.theaters);  
 }  
 })  
 }  
  
 fetchItems();  
 }, []);  
  
   
  
 if(auth){  
 return (  
 <>  
 <CustomerNavbar/>  
   
 {  
 theaters.map(theater => <div key={theater.\_id} style={{margin: 30}}>  
 <ModalTitle>{theater.name}</ModalTitle>  
 <AllMovies theater={theater}/>  
 </div>  
   
 )  
 }  
 </>  
 )  
 }  
   
 return <h1>Unauthorized User</h1>  
}

HomeMovieCard.jsx

import { *useState*, *useEffect* } from "react";  
import { Card, Button, ButtonGroup } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
  
export default function *HomeMovieCard*(props) {  
 const {id} = props;  
  
 const [movie, setMovie] = *useState*();  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/movies/"+id)  
 .then(response => response.json())  
 .then(items => {  
 setMovie(items.movie);  
 });  
 }  
 fetchItems();  
 }, []);  
  
 async function addToCart(){  
 await *fetch*('http://localhost:5000/api/cart', {  
 method: "POST",  
 headers: {  
 'Content-Type': 'application/json',  
 'Authorization' : localStorage.getItem('token'),  
 },  
 body: JSON.stringify({  
 user\_id: localStorage.getItem("user\_id"),  
 movie: movie  
 })  
 }).then(*alert*("Item Added"));  
 }  
  
 if(movie){  
 return <div>  
 {  
 <Card key={movie.banner} style={{ width: '18rem', margin: 5}}>  
 <Card.Img variant="top" src={movie.banner} style={{height:'15rem'}} />  
 <Card.Body style={{textAlign:"center"}}>  
 <Card.Title>{movie.title}</Card.Title>  
 <Card.Text>  
 {movie.description}  
 </Card.Text>  
 <Card.Text>  
 {movie.cast}  
 </Card.Text>  
 <Card.Text>  
 <b>Time:</b> {movie.time}  
 </Card.Text>  
 <Card.Text>  
 <b>Ticket Price:</b> Rs.{movie.price}  
 </Card.Text>  
 <Button onClick={addToCart}>Add to Cart</Button>  
 </Card.Body>  
 </Card>  
 }  
   
 </div>  
  
 }  
  
   
}

Tickets.jsx

import { Table } from "react-bootstrap";  
import {*useEffect*, *useState*} from "react";  
import *CustomerNavbar* from "../navs/CustomerNavbar";  
  
export default function *Tickets*(){  
 const [tickets, setTickets] = *useState*([]);  
 const [auth, setAuth] = *useState*(false);  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/tickets/"+localStorage.getItem('user\_id'), {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(items => {  
 if(items.status) {  
 setAuth(true);  
 setTickets(items.tickets);  
 }  
 })  
 }  
  
 fetchItems();  
   
 }, []);  
 let url ="http://localhost:5000/api/tickets/qr/";  
  
 if(auth){  
 return (  
 <>  
 <CustomerNavbar/>  
 <Table striped bordered hover size="sm" style={{width:400, margin:50}}>  
 <thead>  
 <tr>  
 <th>Movie</th>  
 <th>Ticket</th>  
 </tr>  
 </thead>  
 <tbody>   
 {  
 tickets.map(ticket => <tr key={ticket.\_id}>  
 <td>{ticket.movie.title}</td>  
 <td><a href={url+(ticket.qr\_code)} target={"\_blank"}>ticket</a></td>  
 </tr>)  
 }  
 </tbody>  
   
 </Table>  
   
 </>  
 )  
 }  
   
 return <h1>Unauthorized User</h1>  
}

Payment

Payment.jsx

import React, { *useEffect*, *useState*, *useRef* } from "react";  
import { Form, Button } from "react-bootstrap";  
import { useNavigate } from "react-router-dom";  
  
export default function Payments(){  
 const [items, setItems] = *useState*([]);  
  
 const [total, setTotal] = *useState*(0);  
 const [card, setCard] = *useState*("");  
 const [exp, setExp] = *useState*("");  
 const [cvv, setCVV] = *useState*("");  
  
 const navigate = useNavigate();  
  
 function payNow(e) {  
 e.preventDefault();  
  
 *fetch*("http://localhost:8000/payment", {  
 method: "POST",  
 headers : {  
 'Content-Type': 'application/json',  
 },  
 body: JSON.stringify({  
 card: card,  
 exp: exp,  
 cvv: cvv  
 })  
 }).then(response => response.json())  
 .then(payment => {  
 if(payment.status){  
 *fetch*("http://localhost:5000/api/payments", {  
 method: "POST",  
 headers: {  
 'Content-Type': 'application/json',  
 'Authorization' : localStorage.getItem('token'),  
 },  
 body: JSON.stringify({  
 user\_id: localStorage.getItem('user\_id'),  
 total: total,  
 movies: items,  
 })  
 });  
 *alert*("Payment Successfull");  
 navigate('/cart');  
 }  
 else{  
 *alert*("Invalid Payment Details!");  
 }  
 })   
 }  
  
 function cancelPayment() {  
 navigate('/cart');  
 }  
  
 *useEffect*(() => {  
 async function fetchItems() {  
 await *fetch*("http://localhost:5000/api/cart/"+localStorage.getItem('user\_id'), {  
 headers: {  
 'Authorization' : localStorage.getItem('token')  
 }  
 })  
 .then(response => response.json())  
 .then(item => {  
 let \_total = 0;  
 setItems(item.movie);  
 item.movie.map(movie => {  
 \_total = \_total + *parseInt*(movie.movie.price);  
 setTotal(\_total);  
 })  
 })  
 }  
  
 fetchItems();  
 }, []);  
  
   
  
 return <div>  
 <Form style={{width: 500, margin:30}} onSubmit={payNow}>  
 <h1>Payments</h1>  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Card Number</Form.Label>  
 <Form.Control  
 required  
 maxLength={16}  
 onChange={(e) => setCard(e.target.value)}  
 type="text"  
 value={card}  
 placeholder="Card Number"  
 />  
 </Form.Group>  
  
 <Form.Group className="mb-3" controlId="formUsername">  
 <Form.Label>Expiration Date</Form.Label>  
 <Form.Control  
 required  
 maxLength={5}  
 onChange={(e) => setExp(e.target.value)}  
 type="text"  
 value={exp}  
 placeholder="MM/YY"  
 />  
 </Form.Group>  
 <Form.Group className="mb-3" controlId="formCVV">  
 <Form.Label>CVV Code</Form.Label>  
 <Form.Control  
 required  
 maxLength={3}  
 onChange={(e) => setCVV(e.target.value)}  
 type="number"  
 value={cvv}  
 placeholder="CVV"  
 />  
 </Form.Group>  
 <h2>Total : Rs.{total}</h2>  
 <Button variant="primary" type="submit">Pay Now</Button>  
 <Button onClick={cancelPayment} style={{margin:10}}>Cancel Payment</Button>  
 </Form>  
 </div>  
}