**Name: Aditya Taware**

**Div: TY-C**

**Roll no: 38**

**------------------------------------------------------------------------------------------------**

**Design and develop a responsive website for an online book store using  REACT, Springboot and MySQL having  1) Home Page2) Login Page 3) Catalogue Page: 4) Registration Page: (database)**

**Spring Boot –**

**Controller :**

package com.example.MovieApis.movie;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.RestController;

import com.example.MovieApis.movie.Repository.UserInfoRepository;

import com.example.MovieApis.movie.model.TrialModel;

import com.example.MovieApis.movie.model.UserEntity;

import com.example.MovieApis.movie.model.UserInfo;

@RestController

@CrossOrigin

public class Controller {

    @Autowired

    UService service;

    @Autowired

    UserService userService ;

    //UserInfoRepository userInfoRepository ;

    //MovieRepository MovieRepository ;

    @GetMapping("/getMovies")

    public Object getMovies()

    {

        return service.getMovies() ;

    }

    @GetMapping("/hello")

    public String hello(@RequestParam(value = "name", defaultValue = "World") String name) {

      return String.format("Hello %s!", name);

    }

    @PostMapping("/PostCheck")

        public TrialModel PostCheck(@RequestBody TrialModel movie)

        {

            return movie;

        }

    @PostMapping("/SaveUser")

    public UserEntity saveUser(@RequestBody UserEntity  u)

    {   System.out.println(u.getBook\_name());

        System.out.println(u.getId());

        System.out.println(u);

        //MovieRepository.save(u);

        return service.save(u);

    }

    @PostMapping("/SaveCredential")

    public UserInfo saveCredential(@RequestBody UserInfo  u)

    {

        System.out.println(u.getName());

        System.out.println(u.getPassword());

        System.out.println(u.toString()) ;

         //userInfoRepository.save(u);

        //return u ;

        return service.saveCredentials(u);

    }

    @PostMapping("/CheckAccess")

    public boolean checkAccess(@RequestBody UserInfo u)

    {

        return service.login(u.getName() ,u.getPassword()) ;

    }

  // @DeleteMapping

  // @PutMapping

}

**Service :**

package com.example.MovieApis.movie;

import java.util.ArrayList;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.example.MovieApis.movie.Repository.UserEntityRepository;

import com.example.MovieApis.movie.Repository.UserInfoRepository;

import com.example.MovieApis.movie.model.TrialModel;

import com.example.MovieApis.movie.model.UserEntity;

import com.example.MovieApis.movie.model.UserInfo;

@Service

public class UService {

    @Autowired

    UserEntityRepository movieRepository ;

    @Autowired

    UserInfoRepository userInfoRepository ;

    public Object getMovies(){

        List<TrialModel> movies = new ArrayList<>() ;

        movies.add(new TrialModel("Mirage","" , 9.8 , ""));

        movies.add(new TrialModel("avenger","",8.2,""));

        return movies ;

    }

    public UserEntity save(UserEntity u ){

        movieRepository.save(u);

        return u;

    }

    public UserInfo saveCredentials(UserInfo u ){

        userInfoRepository.save(u) ;

        return u ;

    }

    public boolean login(String name  , String Password)

    {

        Iterable<UserInfo> list = userInfoRepository.findAll();

        for( UserInfo user :list )

        {

            if(user.getName().equals(name) && user.getPassword().equals(Password)){

                return  true ;

            }

        }

        return false ;

    }

}

**UserEntity Repository:**

package com.example.MovieApis.movie.Repository;

import org.springframework.data.repository.CrudRepository;

import com.example.MovieApis.movie.model.UserEntity;

public interface UserEntityRepository extends CrudRepository<UserEntity, Long> {

}

**UserInfo Repository:**

package com.example.MovieApis.movie.Repository;

import org.springframework.data.repository.CrudRepository;

import com.example.MovieApis.movie.model.UserInfo;

public interface UserInfoRepository extends CrudRepository<UserInfo, Long> {

}

**UserEntity Model:**

package com.example.MovieApis.movie.model;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class UserEntity {

    @Id

    @GeneratedValue(strategy=GenerationType.IDENTITY)

    private long id ;

    //@Column

    private String Book\_name ;

    public String getBook\_name(){

        return Book\_name ;

    }

    public void setBook\_name(String Book\_name){

        this.Book\_name = Book\_name ;

    }

    public long getId() { return id ;}

    public void setId(long Id) { this.id = Id ; }

}

**UserInfo model**

package com.example.MovieApis.movie.model;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class UserInfo {

    @Id

    @GeneratedValue(strategy=GenerationType.IDENTITY)

    private long id ;

    //@Column

    private String Name ;

    private String Password;

    public String getName(){

        return Name ;

    }

    public void setName(String Name){

        this.Name = Name ;

    }

    public String getPassword(){

        return Password ;

    }

    public void setPassword(String Password){

        this.Password = Password ;

    }

    public long getId() { return id ;}

    public void setId(long Id) { this.id = Id ; }

    @Override

    public String toString()

    {

        return this.Name +' '+ this.Password ;

    }

}

**React:**

**App.js**

import logo from './logo.svg';

import Header from './Header';

import Content from './Content';

import LoginPage from './LoginPage' ;

import './App.css';

import { BrowserRouter,Routes,Route } from 'react-router-dom';

function App() {

  return (

    <div className="App">

      <BrowserRouter>

        <Routes>

          <Route path="/" element={<LoginPage/>} />

          <Route path="/content" element={<Content/>} />

        </Routes>

      </BrowserRouter>

      {/\* <Header/> \*/}

      {/\* <Content/> \*/}

    </div>

  );

}

export default App;

**LoginPage.js :**

import React,{useEffect, useState} from 'react';

import { useNavigate } from 'react-router-dom';

function LoginPage(){

    const [name,setEmail] = useState();

    const [password,setPassword] = useState();

    const navigate = useNavigate("");

    // useEffect(() => {

    //     const auth = localStorage.getItem("user");

    //     if(auth) {

    //         navigate('/');

    //     }

    // },[]);

    function Email(event){

        setEmail(event.target.value);

    }

    function Password(event){

        setPassword(event.target.value);

    }

    async function handleLogin(){

        let result = await fetch("http://localhost:8080/CheckAccess",{

            method:"post",

            body:JSON.stringify({name,password}),

            headers:{'Content-Type': 'application/json'}

        });

        result = await result.json();

        console.warn(result);

        if(result){

            // navigate("/");

            //alert("All Ok");

            navigate("/content")

        }else{

            alert("Please enter correct details");

        }

    }

    return (

        <div className='login'>

            <h1>Login</h1>

            <input

                type="email"

                className='inputBox'

                placeholder="Email"

                value={name}

                onChange={Email}

             />

            <input

                type="password"

                className='inputBox'

                placeholder="Password"

                value={password}

                onChange={Password}

            />

            <button onClick={handleLogin} className="button">Login</button>

        </div>

    );

}

export default LoginPage;

**Content.js**

import {useState} from 'react' ;

const Content  =  () => {

   const[items, setItems] = useState([

    {

        id : 1 ,

        item : "book1",

        checked : false

    },

    {

        id : 2 ,

        item : "book2",

        checked : false

    },

    {

        id : 3 ,

        item : "book3",

        checked : false

    }

   ])  ;

   const handleCheck= (id) => {

       const newList = items.map((item)=> item.id === id ? {...item , checked: !item.checked} : item)

       setItems(newList)

   }

   const handleSubmit= () => {

    var s = ""

    var val = items.map((item => item.checked ? s+=item.item:s+='')) ;

    const object = {id : 1, book\_name: val[2]} ;

    console.log(object) ;

    fetch("http://localhost:8080/SaveUser" , {

        method: "POST",

        headers:{"Content-Type":"application/json"},

        body : JSON.stringify(object)

    }).then(()=>{

        console.log("Succeed") ;

    })

   }

    return(

        <main>

            <ul>

                {items.map((item)=> (

                    <li className="item" key={item.id}>

                        <input type="checkbox" checked = {item.checked} onChange={()=> handleCheck(item.id)}></input>

                        <label>{item.item}</label>

                    </li>

                ))

                }

            </ul>

            {/\* <input type="button" ></input> \*/}

            <button className="contentButton" onClick={handleSubmit}>Submit</button>

        </main>

    )

}

export default Content ;

**Output**

**Login Page:**

**Graphical user interface

Description automatically generated**

**Registration Page:**

**Graphical user interface, application

Description automatically generated**

**User Entity:**

Graphical user interface, application

Description automatically generated

User Info:

Graphical user interface, table

Description automatically generated