

## Assignment-6

**Name:** Aditya Gorakh Zite

**Div:** TY-CS-D

**Roll No.:** 87

**PRN:** 12211040

### controller/AuthController.java

```
package com.example.backend.controller;

import com.example.backend.model.User;
import com.example.backend.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

@RestController
@CrossOrigin(origins = "*")
@RequestMapping("/api/auth")
public class AuthController {

    @Autowired
    private UserService userService;

    @PostMapping("/register")
    public User register(@RequestBody User user) {
        return userService.register(user);
    }
}
```

```
@PostMapping("/login")
public User login(@RequestBody User user) {
    return userService.login(user.getEmail(), user.getPassword());
}
}
```

#### **controller/ProductController.java**

```
package com.example.backend.controller;

import com.example.backend.model.Product;
import com.example.backend.service.ProductService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@CrossOrigin(origins = "*")
@RequestMapping("/api/products")
public class ProductController {

    @Autowired
    private ProductService productService;

    @GetMapping
    public List<Product> getAll() {
        return productService.getAll();
    }
}
```

```
}
```

```
@PostMapping
```

```
public Product create(@RequestBody Product product) {  
    return productService.create(product);  
}
```

```
@PutMapping("/{id}")
```

```
public Product update(@PathVariable Long id, @RequestBody Product product) {  
    return productService.update(id, product);  
}
```

```
@DeleteMapping("/{id}")
```

```
public boolean delete(@PathVariable Long id) {  
    return productService.delete(id);  
}
```

```
@GetMapping("/search")
```

```
public List<Product> search(@RequestParam String keyword) {  
    return productService.search(keyword);  
}  
}
```

### **model/Product.java**

```
package com.example.backend.model;
```

```
import jakarta.persistence.*;
```

@Entity

```
public class Product {
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Long id;
```

```
    private String name;
```

```
    private String price;
```

```
    // Getters and Setters
```

```
    public Long getId() {
```

```
        return id;
```

```
    }
```

```
    public void setId(Long id) {
```

```
        this.id = id;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
```

```
    }
```

```
    public void setName(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    public String getPrice() {
```

```
        return price;
    }

    public void setPrice(String price) {
        this.price = price;
    }
}
```

### **model/User.java**

```
package com.example.backend.model;

import jakarta.persistence.*;

@Entity
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;
    private String email;
    private String password;

    // Getters and Setters
    public Long getId() {
        return id;
    }
}
```

```
public void setId(Long id) {  
    this.id = id;  
}
```

```
public String getName() {  
    return name;  
}
```

```
public void setName(String name) {  
    this.name = name;  
}
```

```
public String getEmail() {  
    return email;  
}
```

```
public void setEmail(String email) {  
    this.email = email;  
}
```

```
public String getPassword() {  
    return password;  
}
```

```
public void setPassword(String password) {  
    this.password = password;  
}
```

```
}
```

### **repository/ProductRepository.java**

```
package com.example.backend.repository;
```

```
import com.example.backend.model.Product;
```

```
import org.springframework.data.jpa.repository.JpaRepository;
```

```
import java.util.List;
```

```
public interface ProductRepository extends JpaRepository<Product, Long> {
```

```
    List<Product> findByNameContainingIgnoreCase(String keyword);
```

```
}
```

### **repository/UserRepository.java**

```
package com.example.backend.repository;
```

```
import com.example.backend.model.User;
```

```
import org.springframework.data.jpa.repository.JpaRepository;
```

```
public interface UserRepository extends JpaRepository<User, Long> {
```

```
    User findByEmail(String email);
```

```
}
```

## **service/ProductService.java**

```
package com.example.backend.service;
```

```
import com.example.backend.model.Product;
```

```
import com.example.backend.repository.ProductRepository;
```

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

```
import org.springframework.stereotype.Service;
```

```
import java.util.List;
```

```
import java.util.Optional;
```

```
@Service
```

```
public class ProductService {
```

```
    @Autowired
```

```
    private ProductRepository productRepository;
```

```
    public List<Product> getAll() {
```

```
        return productRepository.findAll();
```

```
    }
```

```
    public Product create(Product product) {
```

```
        return productRepository.save(product);
```

```
    }
```

```
    public Product update(Long id, Product updatedProduct) {
```

```
        Optional<Product> optionalProduct = productRepository.findById(id);
```

```
        if (optionalProduct.isPresent()) {
```



```

        Product product = optionalProduct.get();
        product.setName(updatedProduct.getName());
        product.setPrice(updatedProduct.getPrice());
        return productRepository.save(product);
    }
    return null;
}

```

```

public boolean delete(Long id) {
    if (productRepository.existsById(id)) {
        productRepository.deleteById(id);
        return true;
    }
    return false;
}

```

```

public List<Product> search(String keyword) {
    return productRepository.findByNameContainingIgnoreCase(keyword);
}
}

```

### **service/UserService.java**

```

package com.example.backend.service;

import com.example.backend.model.User;
import com.example.backend.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;

```

```
import org.springframework.stereotype.Service;
```

```
@Service
```

```
public class UserService {
```

```
    @Autowired
```

```
    private UserRepository userRepository;
```

```
    public User register(User user) {
```

```
        return userRepository.save(user);
```

```
    }
```

```
    public User login(String email, String password) {
```

```
        User user = userRepository.findByEmail(email);
```

```
        return (user != null && user.getPassword().equals(password)) ? user : null;
```

```
    }
```

```
}
```

### **BackendApplication.java**

```
package com.example.backend;
```

```
import org.springframework.boot.SpringApplication;
```

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
```

```
public class BackendApplication {
```

```
    public static void main(String[] args) {
```

```
    SpringApplication.run(BackendApplication.class, args);
  }

}
```

### **components/Catalogue.jsx**

```
import React, { useEffect, useState } from 'react';
import { API } from '../api';

function Catalogue() {
  const [products, setProducts] = useState([]);
  const [keyword, setKeyword] = useState("");
  const [form, setForm] = useState({ id: null, name: "", price: "" });

  const fetchAll = async () => { const res = await API.get('/products'); setProducts(res.data); };
  useEffect(() => { fetchAll(); }, []);

  const handleSearch = async () => {
    if (!keyword) return fetchAll();
    const res = await API.get('/products/search', { params: { keyword } });
    setProducts(res.data);
  };

  const handleDelete = async id => { await API.delete(`/products/${id}`); fetchAll(); };
  const handleEdit = product => setForm(product);
  const handleSubmit = async e => {
    e.preventDefault();
```

```

    form.id ? await API.put(`/products/${form.id}`, form) : await API.post('/products', form);

    setForm({ id: null, name: "", price: "" });

    fetchAll();

  };

  return (
    <div className="container py-5">
      <h2 className="mb-4">Product Catalogue</h2>

      <div className="input-group mb-4">
        <input type="text" className="form-control" placeholder="Search products..."
value={keyword} onChange={e => setKeyword(e.target.value)} />
        <button className="btn btn-outline-secondary"
onClick={handleSearch}>Search</button>
      </div>

      <form onSubmit={handleSubmit} className="mb-5">
        <div className="row g-2">
          <div className="col-md-5">
            <input type="text" className="form-control" placeholder="Name"
value={form.name} onChange={e => setForm({ ...form, name: e.target.value })} />
          </div>
          <div className="col-md-5">
            <input type="text" className="form-control" placeholder="Price"
value={form.price} onChange={e => setForm({ ...form, price: e.target.value })} />
          </div>
          <div className="col-md-2">
            <button type="submit" className="btn btn-primary w-100">{form.id ?
'Update' : 'Add'}</button>
          </div>
        </div>
      </form>
    </div>
  );

```



```

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

function Home() {
  return (
    <div className="py-5 bg-light text-center">
      <div className="container">
        <h1 className="display-4">Welcome to United Dev's Reference Book Store</h1>
        <p className="lead">Your one-stop online store for quality reference books
delivered to your door.</p>
        <Link className="btn btn-primary btn-lg" to="/catalogue">Shop Now</Link>
      </div>
    </div>
  );
}

```

```

export default Home;

```

### **components/Login.jsx**

```

import React, { useState } from 'react';
import { API } from '../api';
import { useNavigate, Link } from 'react-router-dom';

function Login() {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const navigate = useNavigate();

```

```

const handleLogin = async () => {
  try {
    const res = await API.post('/auth/login', { email, password });
    if (res.data) navigate('/catalogue');
    else alert('Invalid credentials');
  } catch {
    alert('Login failed');
  }
};

return (
  <div className="d-flex justify-content-center align-items-center vh-100">
    <div className="card w-25">
      <div className="card-body">
        <h5 className="card-title mb-4 text-center">Login</h5>
        <div className="mb-3">
          <input className="form-control" type="email" placeholder="Email"
value={email} onChange={e => setEmail(e.target.value)} />
        </div>
        <div className="mb-3">
          <input className="form-control" type="password" placeholder="Password"
value={password} onChange={e => setPassword(e.target.value)} />
        </div>
        <button className="btn btn-primary w-100"
onClick={handleLogin}>Login</button>
        <p className="mt-3 text-center">Don't have an account? <Link
to="/register">Register</Link></p>
      </div>
    </div>
  </div>
)

```

```
    </div>
  );
}
```

```
export default Login;
```

### **components/Navbar.jsx**

```
import React from 'react';
import { Link } from 'react-router-dom';

function Navbar() {
  return (
    <nav className="navbar navbar-expand-lg navbar-light bg-light">
      <div className="container">
        <Link className="navbar-brand" to="/">United Dev's Reference Book Store</Link>
        <button className="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarNav">
          <span className="navbar-toggler-icon"></span>
        </button>
        <div className="collapse navbar-collapse" id="navbarNav">
          <ul className="navbar-nav ms-auto">
            <li className="nav-item"><Link className="nav-link"
to="/">Home</Link></li>
            <li className="nav-item"><Link className="nav-link"
to="/login">Login</Link></li>
            <li className="nav-item"><Link className="nav-link"
to="/register">Register</Link></li>
          </ul>
        </div>
      </div>
    </nav>
  );
}
```



```

        </div>
    </div>
</nav>
);
}

export default Navbar;

```

### **components/Register.jsx**

```

import React, { useState } from 'react';
import { API } from '../api';
import { useNavigate, Link } from 'react-router-dom';

function Register() {
    const [name, setName] = useState('');
    const [email, setEmail] = useState('');
    const [password, setPassword] = useState('');
    const navigate = useNavigate();

    const handleRegister = async () => {
        try {
            const res = await API.post('/auth/register', { name, email, password });
            if (res.data) navigate('/login');
        } catch {
            alert('Registration failed');
        }
    };
}

```

```

return (
  <div className="d-flex justify-content-center align-items-center vh-100">
    <div className="card w-25">
      <div className="card-body">
        <h5 className="card-title mb-4 text-center">Register</h5>
        <div className="mb-3">
          <input className="form-control" type="text" placeholder="Name"
value={name} onChange={e => setName(e.target.value)} />
        </div>
        <div className="mb-3">
          <input className="form-control" type="email" placeholder="Email"
value={email} onChange={e => setEmail(e.target.value)} />
        </div>
        <div className="mb-3">
          <input className="form-control" type="password" placeholder="Password"
value={password} onChange={e => setPassword(e.target.value)} />
        </div>
        <button className="btn btn-success w-100"
onClick={handleRegister}>Register</button>
        <p className="mt-3 text-center">Already have an account? <Link
to="/login">Login</Link></p>
      </div>
    </div>
  </div>
);
}

```

```
export default Register;
```

## **api.jsx**

```
import axios from 'axios';

export const API = axios.create({
  baseURL: 'http://localhost:8080/api',
});
```

## **App.css**

```
#root {
  max-width: 1280px;
  margin: 0 auto;
  padding: 2rem;
  text-align: center;
}

.logo {
  height: 6em;
  padding: 1.5em;
  will-change: filter;
  transition: filter 300ms;
}

.logo:hover {
  filter: drop-shadow(0 0 2em #646cfaa);
}

.logo.react:hover {
  filter: drop-shadow(0 0 2em #61dafbaa);
}
```

```
}
```

```
@keyframes logo-spin {  
  from {  
    transform: rotate(0deg);  
  }  
  to {  
    transform: rotate(360deg);  
  }  
}
```

```
@media (prefers-reduced-motion: no-preference) {  
  a:nth-of-type(2) .logo {  
    animation: logo-spin infinite 20s linear;  
  }  
}
```

```
.card {  
  padding: 2em;  
}
```

```
.read-the-docs {  
  color: #888;  
}
```

## **App.jsx**

```
import React from 'react';
```

```
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';  
import Navbar from './components/Navbar';  
import Home from './components/Home';  
import Login from './components/Login';  
import Register from './components/Register';  
import Catalogue from './components/Catalogue';
```

```
function App() {  
  return (  
    <Router>  
      <Navbar />  
      <Routes>  
        <Route path="/" element={<Home />} />  
        <Route path="/login" element={<Login />} />  
        <Route path="/register" element={<Register />} />  
        <Route path="/catalogue" element={<Catalogue />} />  
      </Routes>  
    </Router>  
  );  
}
```

```
export default App;
```

### **index.css**

```
:root {  
  font-family: system-ui, Avenir, Helvetica, Arial, sans-serif;  
  line-height: 1.5;
```

```
font-weight: 400;
```

```
color-scheme: light dark;
```

```
color: rgba(255, 255, 255, 0.87);
```

```
background-color: #242424;
```

```
font-synthesis: none;
```

```
text-rendering: optimizeLegibility;
```

```
-webkit-font-smoothing: antialiased;
```

```
-moz-osx-font-smoothing: grayscale;
```

```
}
```

```
a {
```

```
font-weight: 500;
```

```
color: #646cff;
```

```
text-decoration: inherit;
```

```
}
```

```
a:hover {
```

```
color: #535bf2;
```

```
}
```

```
body {
```

```
margin: 0;
```

```
display: flex;
```

```
place-items: center;
```

```
min-width: 320px;
```

```
min-height: 100vh;
```

```
}
```

```
h1 {  
  font-size: 3.2em;  
  line-height: 1.1;  
}
```

```
button {  
  border-radius: 8px;  
  border: 1px solid transparent;  
  padding: 0.6em 1.2em;  
  font-size: 1em;  
  font-weight: 500;  
  font-family: inherit;  
  background-color: #1a1a1a;  
  cursor: pointer;  
  transition: border-color 0.25s;  
}
```

```
button:hover {  
  border-color: #646cff;  
}
```

```
button:focus,  
button:focus-visible {  
  outline: 4px auto -webkit-focus-ring-color;  
}
```

```
@media (prefers-color-scheme: light) {  
  :root {  
    color: #213547;
```

```
    background-color: #ffffff;
  }
  a:hover {
    color: #747bff;
  }
  button {
    background-color: #f9f9f9;
  }
}
```

### **Main.jsx**

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import 'bootstrap/dist/css/bootstrap.min.css';
import App from './App.jsx'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```



## Output:

United Dev's Reference Book Store

[Home](#) [Login](#) [Register](#)

# Welcome to United Dev's Reference Book Store

Your one-stop online store for quality reference books delivered to your door.

Shop Now

## Product Catalogue

C The Complete Reference

\$1500

Edit

Delete

Java The Complete Reference

\$1800

Edit

Delete

# Product Catalogue

Search

Add

C The Complete Reference

\$1500

Edit

Delete

Java The Complete Reference

\$1800

Edit

Delete

# Product Catalogue

Search

Add

C The Complete Reference

\$1500

Edit

Delete

Java The Complete Reference

\$1800

Edit

Delete

Node.js The Complete Reference

\$2400

Edit

Delete

# Product Catalogue

Search

Update

C The Complete Reference

\$1500

Edit

Delete

Java The Complete Reference

\$1800

Edit

Delete

Node.js The Complete Reference

\$2400

Edit

Delete

# Product Catalogue

Search

Update

C The Complete Reference

\$1500

Edit

Delete

Java The Complete Reference

\$1800

Edit

Delete

Node.js The Complete Reference

\$2400

Edit

Delete

# Product Catalogue

Search

Add

C The Complete Reference

\$1500

Edit

Delete

Java The Complete Reference

\$1800

Edit

Delete

Node.js The Complete Reference

\$2100

Edit

Delete

# Product Catalogue

Search

Add

Node.js The Complete Reference

\$2100

Edit

Delete

# Product Catalogue

Search

Add

C The Complete Reference

\$1500

Edit

Delete

Node.js The Complete Reference

\$2100

Edit

Delete