1. Create-react-app
2. Make component folder and put the Heder , Footer , Products , Rating into component
3. Using a products.js array of producst and display it on product folder
4. Installing react-bootstrap to use custom components
5. Coping the bootswatch template to src folder
6. Installing react-router-dom
7. Route component for SPA , and instead using href use a to
8. We can route the bootstrap route container
9. We use params /product/:id as match.params
10. Fetching product from react useEffect
11. Connect mongodb to atlas and compass
12. Then using mongoose setup a app

OSNOVNI SETUP ZA BACKEND EXPRESS.JS

const express = require('express')

const products = require('./data/products')

const app = express();

app.get('/', (req, res) => {

    res.send('API is running...')

})

app.get('/api/products', (req, res) => {

    res.json(products)

})

app.get('/api/products/:id', (req, res) => {

    const product = products.find(p => p.\_id === req.params.id)

    res.json(product)

})

app.listen(5000, console.log('Server runing on port 5000'))

THINGS LIKE PRODUCTS , USERS , CART IS GLOBAL STATE ,

USING Forms and Inputs and simple stuffs we use a useState

Fetching data from server with axios

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

import axios from 'axios';

import { Row, Col } from 'react-bootstrap';

import Product from '../components/Product';

const HomeScreen = () => {

    const [products, setProducts] = useState([])

    useEffect(() => {

        const fetchProducts = async () => {

            const { data } = await axios.get('/api/products')

            setProducts(data)

        }

        fetchProducts();

    }, [])

    return (

        <>

ENV in global we can put a API keys,global variables

Connecting Mongo Db and Node.js server

import mongoose from 'mongoose'

const connectDB = async () => {

    try {

        const conn = await mongoose.connect(process.env.MONGO\_URI, {

            useUnifiedTopology: true,

            useNewUrlParser: true,

            useCreateIndex: true

        })

        console.log(`MongoDB Connected: ${conn.connection.host}`)

    } catch (error) {

        console.error(`Error:${error.message}`)

        process.exit(1);

    }

}

export default connectDB;

and import to server.js

installing colors to node.js server

Creating Mongoose schema

import mongoose from 'mongoose'

const userSchema = mongoose.Schema({

    name: {

        type:String,

        required:true

    },

    email: {

        type:String,

        required:true,

        unique: true

    },

    password: {

        type:String,

        required:true

    },

    isAdmin: {

        type:Boolean,

        required:true,

        default: false

    }

}, {

    timestamps: true

})

const User = mongoose.model('User', userSchema)

export default User

 user: {

        type:mongoose.Schema.Types.ObjectId,

        required:true,

        ref: 'User'

    },

This define connect between product and user

Product schema

import mongoose from 'mongoose';

const reviewSchema = mongoose.Schema({

    name: { type: String, required: true},

    rating: { type: Number, required: true},

    comment: { type: String, required: true},

}, {timestamps: true})

const productSchema = mongoose.Schema({

    user: {

        type:mongoose.Schema.Types.ObjectId,

        required:true,

        ref: 'User'

    },

    name: {

        type:String,

        required:true

    },

    image: {

        type:String,

        required:true

    },

    brand: {

        type:String,

        required:true

    },

    category: {

        type:String,

        required:true

    },

    description: {

        type:String,

        required:true

    },

    reviews: [reviewSchema],

    rating: {

        type: Number,

        required: true,

        default: 0

    },

    numReviews: {

        type: Number,

        required: true,

        default:0

    },

    price: {

        type: Number,

        required: true,

        default: 0

    },

    countInStock: {

        type: Number,

        required: true,

        default: 0

    },

}, {

    timestamps: true

})

const Product = mongoose.model('Product', productSchema)

export default User

import mongoose from 'mongoose';

const orderSchema = mongoose.Schema({

    name: { type: String, required: true},

    rating: { type: Number, required: true},

    comment: { type: String, required: true},

}, {timestamps: true})

const productSchema = mongoose.Schema({

    user: {

        type:mongoose.Schema.Types.ObjectId,

        required:true,

        ref: 'User'

    },

    orderItems: [

        {

            name: { type: String, required: true},

            qty: { type: String, required: true},

            image: { type: String, required: true},

            price: { type: Number, required: true},

            product: {

                type: mongoose.Schema.Types.ObjectId,

                required: true,

                ref: 'Product'

            },

        }

    ],

    shippingAddress: {

        address: { type: String, required: true},

        city: { type: String, required: true},

        postalCode: { type: String, required: true},

        country: { type: String, required: true}

    },

    paymentResult: {

        id: { type: String},

        status: { type: String},

        update\_time: { type: String},

        email\_address: { type: String}

    },

    taxPrice: {

        type: Number,

        required: true,

        default: 0.0

    },

    shippingPrice: {

        type: Number,

        required: true,

        default: 0.0

    },

    totalPrice: {

        type: Number,

        required: true,

        default: 0.0

    },

    isPaid: {

        type: Boolean,

        required: true,

        default: false

    },

    paidAt: {

        type: Date,

    },

    isDelivered: {

        type: Boolean,

        required: true,

        default: false

    },

    deliveredAt: {

        type: Date,

    },

}, {

    timestamps: true

})

const Order = mongoose.model('Order', OrderSchema)

export default User

after setup datebase we create a models

        const adminUser = createdUsers[0].\_id

first user from data users

export module uvjek stavljamo na odredjenu funkciju

import express from 'express';

import dotenv from 'dotenv';

import colors from 'colors';

import connectDB from './config/db.js';

import productRoutes from './routes/productRoutes.js';

dotenv.config();

connectDB();

const app = express();

app.get('/', (req, res) => {

    res.send('API is running...')

})

app.use('/api/products', productRoutes)

const PORT = process.env.PORT || 5000

app.listen(

    PORT,

    console.log(`Server running in ${process.env.NODE\_ENV} mode on port ${PORT}`.yellow.bold)

)

import express from 'express'

import asyncHandler from 'express-async-handler'

const router = express.Router();

import Product from '../models/productModel.js'

// @desc Fetch all products

// @route GET /api/products

//@access Public

router.get('/', asyncHandler(async (req, res) => {

    const products = await Product.find({})

    res.json(products)

}))

// @desc Fetch single product

// @route GET /api/products/:id

//  @access Public

router.get('/:id', asyncHandler(async (req, res) => {

    const product = await Product.findById(req.params.id)

    if(product) {

        res.json(product)

    } else {

        res.status(404).json({ message: 'Product not found'})

    }

    res.json(product)

}))

export default router

const ProductScreen = ( { match }) => {

    const [product, setProduct] = useState({})

    useEffect(() => {

        const fetchProduct = async () => {

            const { data } = await axios.get(`/api/products/${match.params.id}`)

            setProduct(data)

        }

        fetchProduct();

    }, [match])

 return (

        <>

            <Link className='btn btn-light mb-3' to='/'>

                Go Back

            </Link>

            <Row>

                <Col md={6}>

                    <Image src={product.image} alt={product.name} fluid/>

                </Col>

                <Col md={3}>

                    <ListGroup variant='flush'>

                        <ListGroup.Item>

                            <h3>{product.name}</h3>

                        </ListGroup.Item>

                        <ListGroup.Item>

                            <Rating value={product.rating} text={`${product.numReviews} reviews`}/>

                        </ListGroup.Item>

                    </ListGroup>

                    <ListGroup.Item>

                        Price: ${product.price}

                    </ListGroup.Item>

                    <ListGroup.Item>

                        Description: ${product.description}

                    </ListGroup.Item>

                </Col>

                <Col md={3}>

                    <Card>

                        <ListGroup variant='flush'>

                            <ListGroup.Item>

                                <Row>

                                    <Col>

                                        Price:

                                    </Col>

                                    <Col>

                                        <strong>${product.price}</strong>

                                    </Col>

                                </Row>

                            </ListGroup.Item>

                            <ListGroup.Item>

                                <Row>

                                    <Col>

                                        Status:

                                    </Col>

                                    <Col>

                                        {product.countInStock > 0 ? 'In Stock' : 'Out Of Stock'}

                                    </Col>

                                </Row>

                            </ListGroup.Item>

                            <ListGroup.Item>

                                <Button className='btn-block' type='button' disabled={product.countInStock === 0}>

                                    Add to Cart

                                </Button>

                            </ListGroup.Item>

                        </ListGroup>

                    </Card>

                </Col>

            </Row>

        </>

    )

}

export default ProductScreen

const ProductScreen = ( { history, match }) => {

    const [qty, setQty] = useState(0)

    const dispatch = useDispatch()

    useEffect(() => {

        dispatch(listProductDetails(match.params.id))

    }, [match])

    const productDetails = useSelector(state => state.productDetails)

    const { loading, error, product } = productDetails

    const addToCartHandler = () => {

        history.push(`/cart/${match.params.id}?qty=${qty}`)

    }

         {product.countInStock > 0 && (

                            <ListGroup.Item>

                                <Row>

                                    <Col>Qty</Col>

                                    <Col>

                                        <Form.Control as='select' value={qty} onChange={(e) => setQty(e.target.value)} >

                                           {[...Array(product.countInStock).keys()].map(x => (

                                               <option key={x + 1} value={x + 1}>{x + 1}</option>

                                           )) }

                                        </Form.Control>

                                    </Col>

                                </Row>

                            </ListGroup.Item>

                        )}

                        <ListGroup.Item>

                            <Button onClick={addToCartHandler}

                                    className='btn-block'

                                    type='button'

                                    disabled={product.countInStock === 0}>

                                Add to Cart

                            </Button>

                        </ListGroup.Item>

46fSsRSzyW2YgnbqFvDDGXe4rgQ5wDJd2eTEAtkrsDuCcPwz3Hp5YnkcMizAA3MBeahwiYStAArcCF5TYJxUr9N2STJqwND