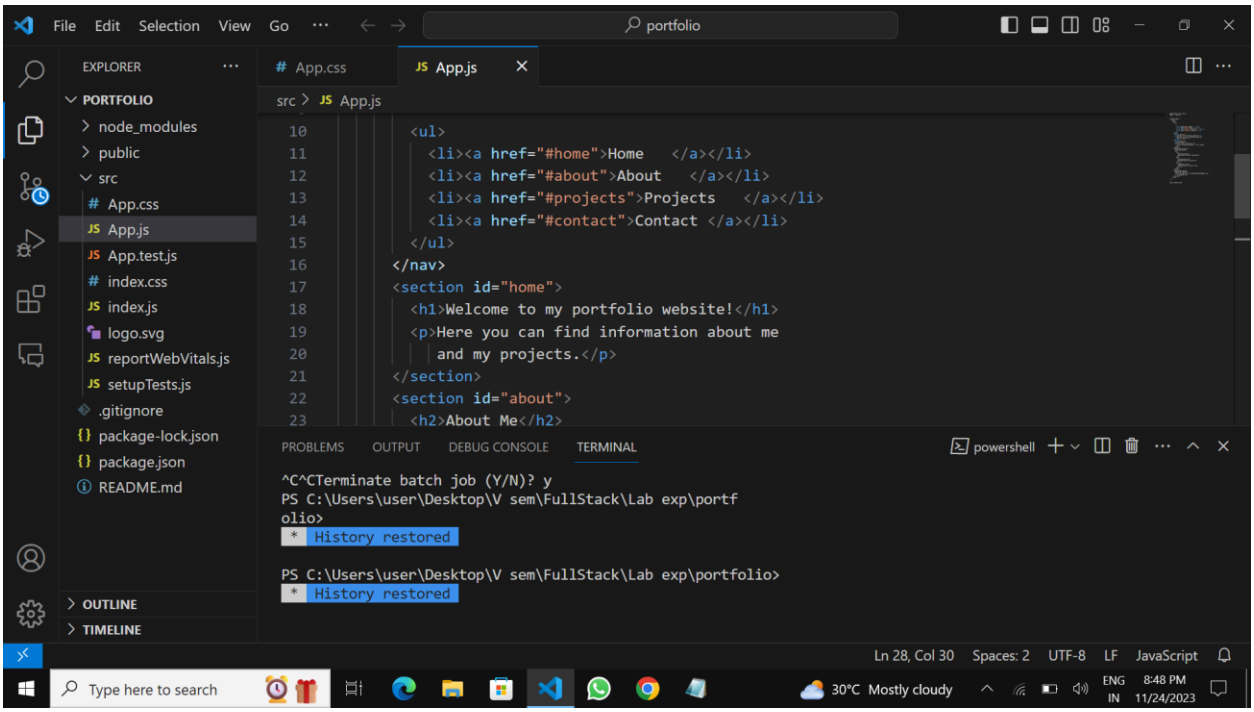
```jsx
import React from 'react';
import './App.css';


function App() {
  return (
    <div>
      <nav>

        <ul>
          <li><a href="#home">Home   </a></li>
          <li><a href="#about">About   </a></li>
          <li><a href="#projects">Projects   </a></li>
          <li><a href="#contact">Contact </a></li>
        </ul>
      </nav>
      <section id="home">
        <h1>Welcome to my portfolio website!</h1>
        <p>Here you can find information about me
           and my projects.</p>
      </section>
      <section id="about">
        <h2>About Me</h2>
        <p>Hi, my name is ABC and I'm a
           junior FullStack developer. I am a fresher
        .</p>
      </section>
      <section id="projects">
        <h2>Projects</h2>
        <ul>
          <li>
            <h3>Portfolio</h3>
            <p>personal Portfolio website</p>
          </li>
          <li>
            <h3>calculator</h3>
            <p>Simple math calculator</p>
          </li>
          {/* <li>
            <h3>Project 3</h3>
            <p>Description of Project 3</p>
          </li> */}
        </ul>
      </section>
```

```
      <section id="contact">
        <h2>Contact Me</h2>
        <p>You can reach me at FullStackDeveloper@gmail.com
           or 12345678.</p>
      </section>
    </div>
  );
}

export default App;
```



2.

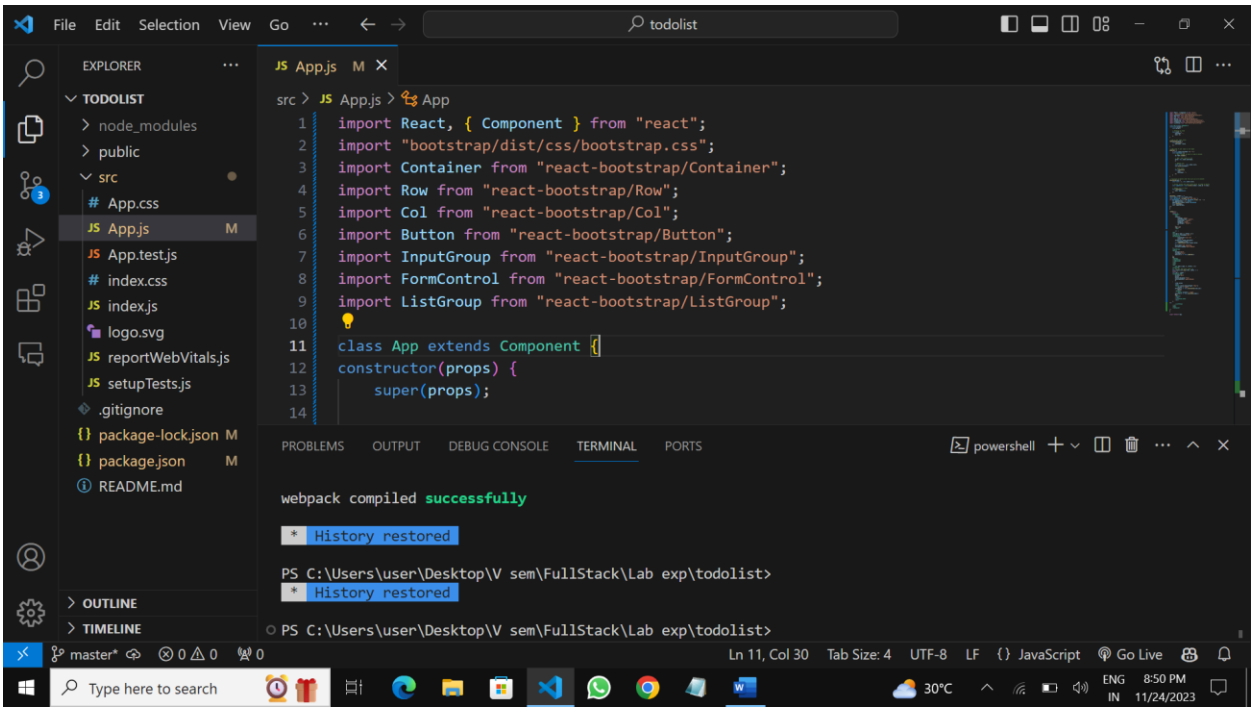
```
import React, { Component } from "react";
import "bootstrap/dist/css/bootstrap.css";
import Container from "react-bootstrap/Container";
import Row from "react-bootstrap/Row";
import Col from "react-bootstrap/Col";
import Button from "react-bootstrap/Button";
import InputGroup from "react-bootstrap/InputGroup";
import FormControl from "react-bootstrap/FormControl";
```

```
import ListGroup from "react-bootstrap/ListGroup";

class App extends Component {
constructor(props) {
    super(props);

    // Setting up state
    this.state = {
        userInput: "",
        list: [],
    };
}

// Set a user input value
updateInput(value) {
    this.setState({
        userInput: value,
    });
}

// Add item if user input in not empty
addItem() {
    if (this.state.userInput !== "") {
    const userInput = {
        // Add a random id which is used to delete
        id: Math.random(),

        // Add a user value to list
        value: this.state.userInput,
        };

        // Update list
        const list = [...this.state.list];
        list.push(userInput);

        // reset state
        this.setState({
            list,
            userInput: "",
        });
    }
}

// Function to delete item from list use id to delete
deleteItem(key) {
```

```
    const list = [...this.state.list];

    // Filter values and leave value which  we need to delete
    const updateList = list.filter((item)=> item.id !== key);

    // Update list in state
    this.setState({
        list: updateList,
    });
}

editItem = (index) => {
const todos = [...this.state.list];
const editedTodo = prompt('Edit the todo:');
if (editedTodo !== null && editedTodo.trim() !== '') {
    let updatedTodos = [...todos]
    updatedTodos[index].value= editedTodo
    this.setState({
    list: updatedTodos,
});
}
}

render() {
    return (
    <Container>
        <Row
        style={{
            display: "flex",
            justifyContent: "center",
            alignItems: "center",
            fontSize: "3rem",
            fontWeight: "bolder",
            }}
        >
        TODO LIST
        </Row>

        <hr />
    <Row>
    <Col md={{ span: 5, offset: 4 }}>
    <InputGroup className="mb-3">
    <FormControl
            placeholder="add item . . . "
            size="lg"
```

```
        value={this.state.userInput}
            onChange={(item) =>
        this.updateInput(item.target.value)
        }
        aria-label="add something"
        aria-describedby="basic-addon2"
    />
    <InputGroup>
    <Button
        variant="dark"
        className="mt-2"
        onClick={() => this.addItem()}
    >
    ADD
    </Button>
    </InputGroup>
    </InputGroup>
    </Col>
    </Row>
    <Row>
        <Col md={{ span: 5, offset: 4 }}>
    <ListGroup>
    {/* map over and print items */}
    {this.state.list.map((item, index) => {
    return (
    <div key = {index} >
    <ListGroup.Item
        variant="dark"
        action
        style={{display:"flex",
        justifyContent:'space-between'
        }}
    >
        {item.value}
        <span>
        <Button style={{marginRight:"10px"}}
            variant = "light"
        onClick={() => this.deleteItem(item.id)}>
            Delete
        </Button>
            <Button variant = "light"
        onClick={() => this.editItem(index)}>
        Edit
        </Button>
        </span>
```

```
        </ListGroup.Item>
        </div>
    );
})}
        </ListGroup>
    </Col>
    </Row>
    </Container>
);
}
}

export default App;
```



3.

App.js

```
import React from "react";
import "./App.css";

import Posts from "./components/Posts";
import Navbar from "./components/BlogNav"

const App = () => {
```

```
return (
    <div className="main-container"
        style={{backgroundColor: "aliceblue"}}>
    <Navbar />
    <Posts />
    </div>
);
};

export default App;
```

BlogNav.js

```
import React from "react";
import 'bootstrap/dist/css/bootstrap.css';
import { Navbar, Nav, Form, FormControl } from 'react-bootstrap';

const BlogNav = () => {
return (
<div>
<Navbar style={{
backgroundColor:"#A3C1D4"
}}>
<img
src='https://media.geeksforgeeks.org/gfg-gg-logo.svg'
height='30'
alt=''
loading='lazy'
/>
<Navbar.Brand href="#home" style={{color:"white",
marginLeft:"10px"}}>study point</Navbar.Brand>
<Navbar.Toggle />
<Navbar.Collapse id="basic-navbar-nav"
className="d-flex justify-content-end">
<Nav>
<Nav.Link href="#home" style={{color:"white"}}>
JavaScript
</Nav.Link>
<Nav.Link href="#about" style={{color:"white"}}>
Data Structure
</Nav.Link>
<Nav.Link href="#services" style={{color:"white"}}>
Algorithm
</Nav.Link>
<Nav.Link href="#contact" style={{color:"white"}}>
```

```
Computer Network
</Nav.Link>
</Nav>
<Form inline>
<FormControl type="text" placeholder="Search"
className="ml-auto" />
</Form>
</Navbar.Collapse>
</Navbar>
</div>
)
}

export default BlogNav;
```

posts.js

```
import React from "react";
import Post1 from "./Post1";
import Post2 from "./Post2";
import Post3 from "./Post3";
import Post4 from "./Post4";
import { Container, Row, Col, Card } from 'react-bootstrap';


const Posts = () => {
return (
<Container>
<Row className="justify-content-between">
    <Col md={8} className="mb-4 mt-4">
    <Post1 />
        </Col>
        <Col md={2} className="mt-4 float-right">
        <Card>
        <Card.Body>
        <Card.Title>Recent Posts</Card.Title>
            <ul className="list-unstyled">
            <li><a href="#">JavaScript</a></li>
            <li><a href="#">Data Structure</a></li>
            <li><a href="#">Algorithm</a></li>
            <li><a href="#">Computer Network</a></li>
            </ul>
        </Card.Body>
        </Card>
        </Col>
```

```
        <Col md={8} className="mb-4">
        <Post2 />
        </Col>
        <Col md={8} className="mb-4">
        <Post3 />
        </Col>
        <Col md={8} className="mb-4">
        <Post4 />
        </Col>
    </Row>
</Container>
);
};

export default Posts;
```

post1.js

```
import { Card } from "react-bootstrap";

const Post1 = () => {
return (
<Card>
<Card.Img
variant="top"
src="https://media.geeksforgeeks.org/wp-content/cdn-
uploads/20230305183140/Javascript.jpg"
width={20}
height={250}
/>
<Card.Body>
<Card.Title>JAVASCRIPT</Card.Title>
<Card.Text>
    JavaScript is the world most popular
    lightweight, interpreted compiled programming
    language. It is also known as scripting
    language for web pages. It is well-known for
    the development of web pages, many non-browser
    environments also use it. JavaScript can be
    used for Client-side developments as well as
    Server-side developments
    </Card.Text>
    <a href="#" className="btn btn-primary">Read More</a>
    </Card.Body>
</Card>
```

```
);
};

export default Post1;
```

post2.js

```
import { Card } from "react-bootstrap";

const Post2 = () => {
return (
        <Card>
    <Card.Img
    variant="top"
    src="https://media.geeksforgeeks.org/img-practice/banner/coa-gate-2022-
thumbnail.png"
    width={20}
    height={250}
    />
    <Card.Body>
    <Card.Title>Data Structure</Card.Title>
    <Card.Text>
        The word Algorithm means “a process
        or set of rules to be followed in calculations
        or other problem-solving operations”. Therefore
        Algorithm refers to a set of rules/instructions
            that step-by-step define how a work is to be
        executed upon in order to get the expected
        results.
    </Card.Text>
    <a href="#" className="btn btn-primary">Read More</a>
    </Card.Body>
    </Card>
)
}

export default Post2;
```

post3.js

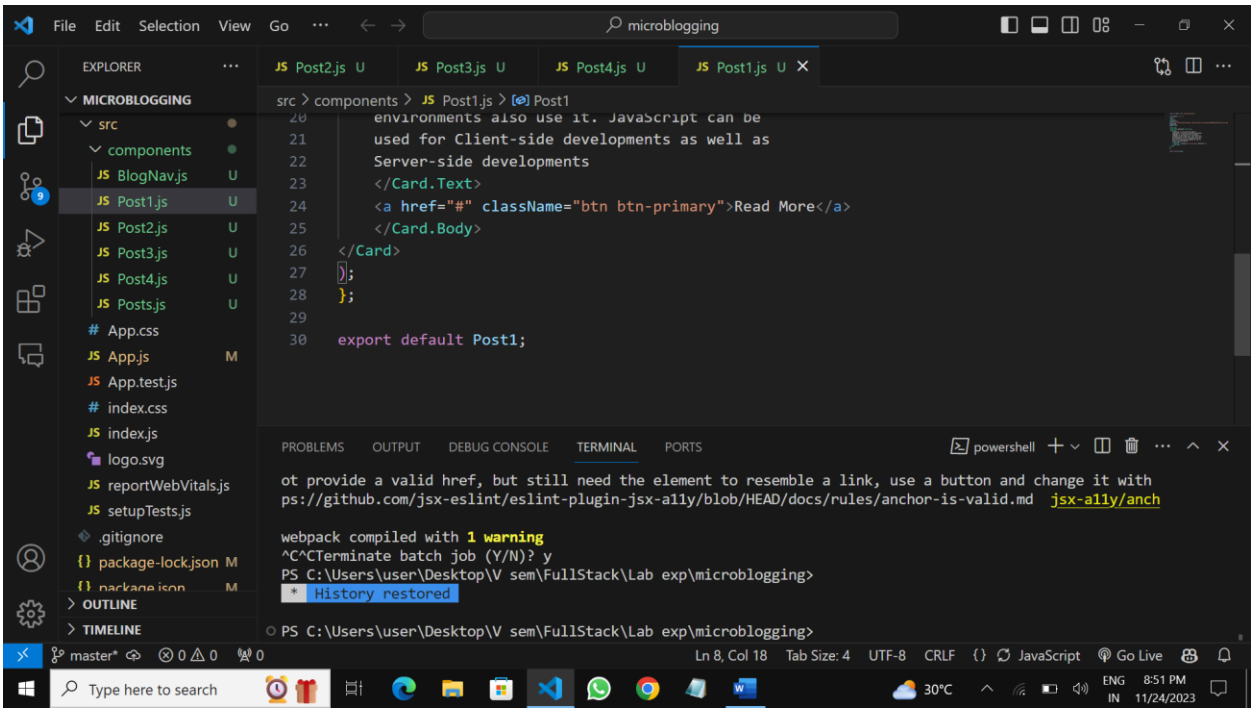
```
import { Card } from "react-bootstrap";
const Post3 = () => {
return (
        <Card>
    <Card.Img
```

```
        variant="top"
    src="https://media.geeksforgeeks.org/img-practice/banner/google-test-series-
thumbnail.png"
    width={20}
    height={250}
        />
    <Card.Body>
    <Card.Title>Algorithm</Card.Title>
    <Card.Text>
        The word Algorithm means “a process
        or set of rules to be followed in calculations
        or other problem-solving operations”. Therefore
        Algorithm refers to a set of rules/instructions
        that step-by-step define how a work is to be
        executed upon in order to get the expected
        results.
    </Card.Text>
    <a href="#" className="btn btn-primary">Read More</a>
    </Card.Body>
    </Card>
)
}

export default Post3;
```

post4.js

```
import { Card } from "react-bootstrap";

const Post4 = () => {
return (
    <Card>
    <Card.Img
    variant="top"
    src="https://media.geeksforgeeks.org/img-practice/banner/cp-maths-java-
thumbnail.png"
        width={20}
    height={250}
    />
        <Card.Body>
    <Card.Title>Computer Network</Card.Title>
    <Card.Text>
        An interconnection of multiple devices,
        also known as hosts, that are connected using
            multiple paths for the purpose of sending/
```

```
        receiving data media. Computer networks can
        also include multiple devices/mediums which
        help in the communication between two different
        devices; these are known as Network devices
        and include things such as routers, switches,
            hubs, and bridges.
                </Card.Text>
        <a href="#" className="btn btn-primary">Read More</a>
        </Card.Body>
    </Card>
)
}

export default Post4;
```



4.>

Index.html

```
 <!DOCTYPE html>

<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
  <link rel="stylesheet" href="styles.css">
  <title>Food Delivery</title>
</head>
<body>
  <header>
    <h1>Food Delivery</h1>
  </header>
  <div class="menu">
    <div class="item">
      <img src="burger.jpg" alt="Burger">
      <h2>Burger</h2>
      <p>Delicious burger with all the fixings.</p>
      <span class="price">$12</span>
      <button class="add-to-cart">Add to Cart</button>
    </div>
    <div class="item">
      <img src="pizza.jpg" alt="Pizza">
      <h2>Pizza</h2>
      <p>Classic cheese pizza with tomato sauce.</p>
      <span class="price">$10</span>
      <button class="add-to-cart">Add to Cart</button>
    </div>
    <!-- Add more items here -->
  </div>
  <div class="cart">
    <h2>Cart</h2>
    <ul id="cart-items"></ul>
    <h3>Total: $<span id="cart-total">0.00</span></h3>
  </div>
  <script src="script.js"></script>
</body>
</html>
```

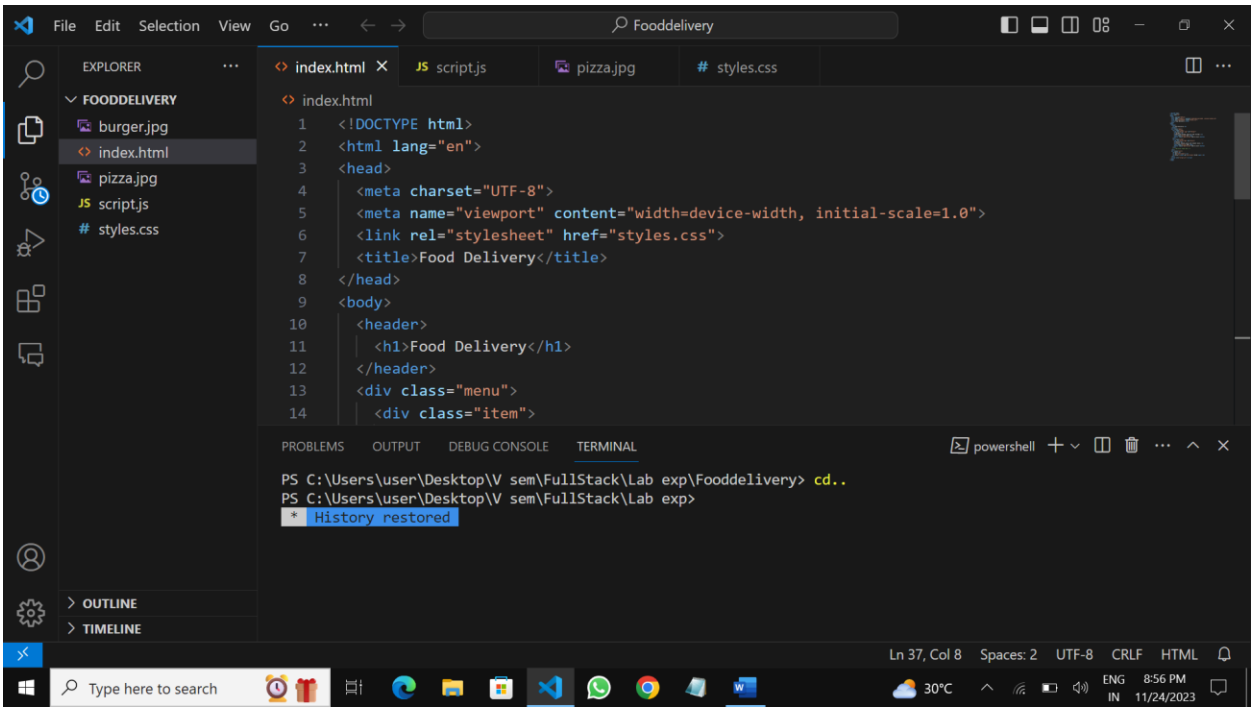
Styles.css

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
  }

  header {
    background-color: #333;
    color: white;
    text-align: center;
```

```
  padding: 10px;
}

.menu {
  display: flex;
  justify-content: space-around;
  flex-wrap: wrap;
  padding: 20px;
}

.item {
  border: 1px solid #ddd;
  padding: 10px;
  margin: 10px;
  text-align: center;
  width: 200px;
}

.item img {
  max-width: 100%;
}

.price {
  font-weight: bold;
}

.add-to-cart {
  background-color: #333;
  color: white;
  border: none;
  padding: 5px 10px;
  cursor: pointer;
}

/* .cart {
  border: 1px solid #ddd;
  padding: 10px;
  position: fixed;
  top: 350px;
  right: 40px;
  width: 250px;
} */
```

Script.js

```
const addToCartButtons = document.querySelectorAll('.add-to-cart');
const cartItemsList = document.getElementById('cart-items');
const cartTotal = document.getElementById('cart-total');

let total = 0;
const cart = {};

addToCartButtons.forEach(button => {
  button.addEventListener('click', () => {
    const item = button.parentElement;
    const itemName = item.querySelector('h2').textContent;
    const itemPrice = parseFloat(item.querySelector('.price')
.textContent.slice(1));

    if (cart[itemName]) {
      cart[itemName] += 1;
    } else {
      cart[itemName] = 1;
    }

    total += itemPrice;
    updateCart();
  });
});

function updateCart() {
  cartItemsList.innerHTML = '';
  for (const item in cart) {
    const li = document.createElement('li');
    li.textContent = `${item} x${cart[item]}`;
    cartItemsList.appendChild(li);
  }
  cartTotal.textContent = total.toFixed(2);
}
```



5.

App.js

```
document.addEventListener('DOMContentLoaded', () => {
    const productList = document.getElementById('product-list');

    // Function to fetch and display product listings
    const fetchProducts = () => {
      fetch('/api/products')
        .then(response => response.json())
        .then(products => {
          productList.innerHTML = ''; // Clear existing list
          products.forEach(product => {
            const li = document.createElement('li');
            li.textContent = `${product.title} - $${product.price}`;

            // Add a "Buy" button for each product
            const actionButton = document.createElement('button');
            if (product.status === 'purchased') {
              actionButton.textContent = 'Purchased';
              actionButton.disabled = true;
            } else {
              actionButton.textContent = 'Buy';
              actionButton.addEventListener('click', () =>buyProduct(product._id,
actionButton));
            }
```

```
            li.appendChild(actionButton);
            productList.appendChild(li);
          });
        })
        .catch(error => console.error('Error fetching products:', error));
    };
    // Call the fetchProducts function initially and after creating a listing
    fetchProducts();

    // Create a new product listing
    const createListingButton = document.getElementById('create-listing-button');
    createListingButton.addEventListener('click', () => {const title =
document.getElementById('title').value;
      const description = document.getElementById('description').value;
      const price = document.getElementById('price').value;

      const newData = {
        title: title,
        description: description,
        price: price
      };

      fetch('/api/products', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify(newData)
      })
        .then(response => response.json())
        .then(product => {
          console.log('Product created:', product);
          // Clear input fields
          document.getElementById('title').value = '';
          document.getElementById('description').value = '';
          document.getElementById('price').value = '';

          // Fetch and display products after creating a listing
          fetchProducts();
        })
        .catch(error => console.error
  ('Error creating product:', error));
    });
  // Buy a product
```

```
    const buyProduct = (productId, actionButton) => {
      fetch(`/api/products/${productId}/buy`, {
        method: 'PUT',
        headers: {
          'Content-Type': 'application/json'
        }
      })
      .then(response => response.json())
      .then(updatedProduct => {
        console.log('Product purchased:', updatedProduct);

        // Update the button text to "Purchased" and disable it
        actionButton.textContent = 'Purchased';
        actionButton.disabled = true;

        // Optionally, you can add aCSS class to style the "Purchased" button
differently
        actionButton.classList.add('purchased-button');
      })
        .catch(error => console.error
  ('Error purchasing product:',
   error));
    };

    // Initial fetch and display of products
    fetchProducts();
  });
```

Index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Classifieds App</title>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
  <h1>Welcome to Classifieds App</h1>



  <h2>Product Listings</h2>
  <ul id="product-list"></ul>

  <h2>Create a Listing</h2>
```

```
  <form id="create-listing-form">
    <label for="title">Title:</label>
    <input type="text" id="title" required><br>
    <label for="description">Description:</label>
    <textarea id="description" required></textarea><br>
    <label for="price">Price:</label>
    <input type="number" id="price" required><br>
    <button type="button" id="create-listing-button">
Create Listing</button>
  </form>

  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <script src="app.js"></script>
</body>

<style>
  body {
    font-family: Arial, sans-serif;
    margin: 20px;
  }

  h1 {
    color: #333;
  }

  h2 {
    color: #555;
    margin-top: 20px;
  }

  ul {
    list-style-type: none;
    padding: 0;
  }

  li {
    margin: 10px 0;
    padding: 10px;
    border: 1px solid #ddd;
    background-color: #f9f9f9;
  }

  label {
    display: inline-block;
    width: 100px;
```

```
  }

  input, textarea {
    width: 300px;
    padding: 5px;
    margin-bottom: 10px;
  }

  button {
    background-color: #333;
    color: #fff;
    padding: 5px 10px;
    border: none;
    cursor: pointer;
  }
  /* Style for the "Purchased" button */
  .purchased-button {
    background-color: #ccc;
    color: #888;
    cursor: not-allowed;
  }

</style>
</html>
```

Models/Product.js

```
const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({
  title: {
    type: String,
    required: true,
  },
  description: {
    type: String,
    required: true,
  },
  price: {
    type: Number,
    required: true,
  },
  status: {
    type: String,
    default: 'available',
```

```
    enum: ['available', 'purchased'],
  },
});

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

module.exports = Product;
```

models/User.js

```
const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({
  username: {
    type: String,
    required: true,
    unique: true,
  },
  email: {
    type: String,
    required: true,
    unique: true,
  },
  password: {
    type: String,
    required: true,
  },
  // Add other user-related fields as needed
});

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

module.exports = User;
```

routes/ProductRoutes.js

```
const express = require('express');
const router = express.Router();
const Product = require('../models/Product.js');

// Create a new product listing
router.post('/', async (req, res) => {
  try {
```

```
    const newProduct = await Product.create(req.body);
    res.json(newProduct);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});

// Get all product listings
router.get('/', async (req, res) => {
  try {
    const products = await Product.find();
    res.json(products);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
// Update product status to mark as purchased
router.put('/:productId/buy', async (req, res) => {
  try {
    const productId = req.params.productId;

    // Find the product by ID and update its status or relevant fields
    const updatedProduct = await Product.findByIdAndUpdate(
      productId,
      { $set: { status: 'purchased' } }, // Updatethe status or relevant fields
here
      { new: true } // Return the updated product
    );

    if (!updatedProduct) {
      return res.status(404).json({ message: 'Product not found' });
    }

    res.json(updatedProduct);
  } catch (error) {
    console.error('Error buying product:', error);
    res.status(500).json({ message: 'An error occurred while buying the product'
});
  }
});
module.exports = router;
```

routes/userRoutes.js

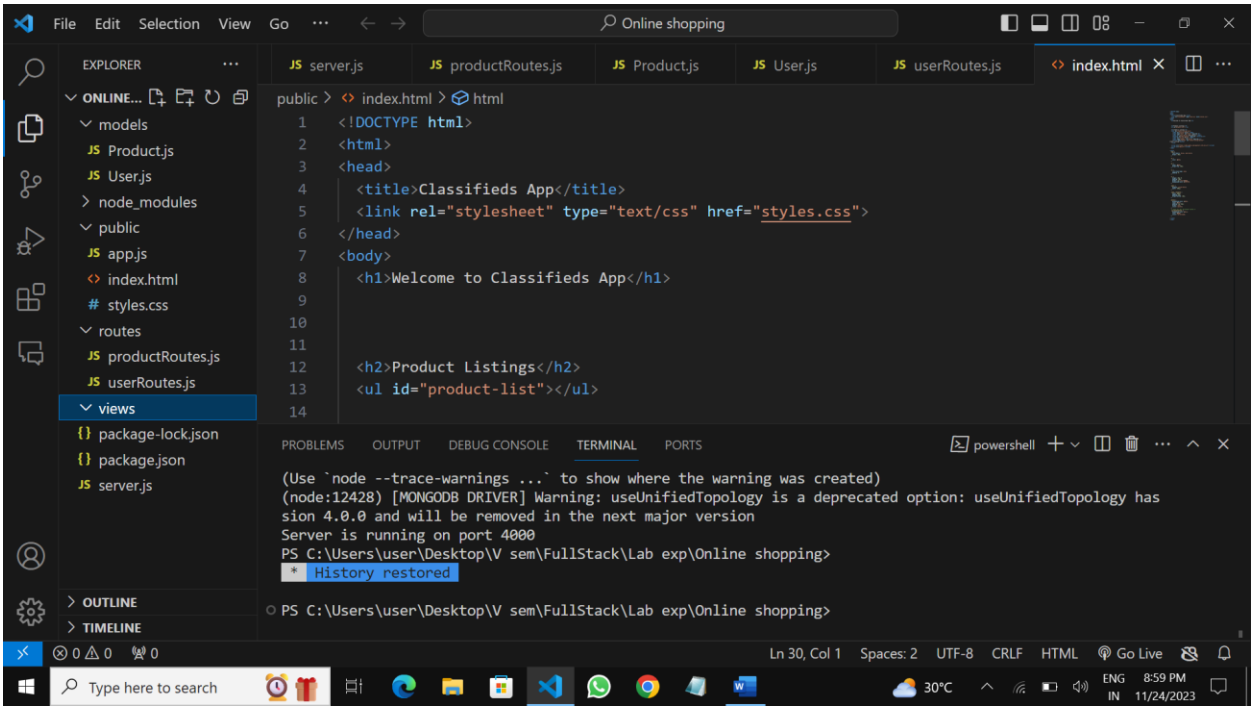
```
const express = require('express');
```

```
const router = express.Router();
const User = require('../models/User');

// User registration
router.post('/register', async (req, res) => {
  try {
    const newUser = await User.create(req.body);
    res.json(newUser);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});

module.exports = router;
```

server.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');
const path = require('path'); // Import the 'path' module

const app = express();


// Middleware
app.use(cors());
app.use(bodyParser.json());

// Connect to MongoDB
// Connect to MongoDB
mongoose.connect('mongodb://127.0.0.1:27017/classifieds', {
  useNewUrlParser: true, // Remove this line (not needed anymore)
  useUnifiedTopology: true, // Remove this line (not needed anymore)
});

mongoose.connection.on('error', console.error.bind(console,
'MongoDB connection error:'));
app.use(express.static(path.join(__dirname, 'public')));
// Routes
const userRoutes = require('./routes/userRoutes.js');
const productRoutes = require('./routes/productRoutes');
app.use('/api/users', userRoutes);
app.use('/api/products', productRoutes);
```

```
app.use(express.static(path.join(__dirname, 'public')));


const PORT = process.env.PORT || 4000;
app.listen(4000, () => {
  console.log(`Server is running on port ${PORT}`);
});
```



6.

Index.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Leave Management System</title>
 <link rel="stylesheet" href="/style.css">
</head>
<body>
 <h1>Welcome to Leave Management System</h1>
 <form action="http://localhost:4000/apply" method="POST">
 <label for="employeeName">Employee Name:</label>
 <input type="text" id="employeeName" name="employeeName" required> <br>
<label for="leaveType">Leave Type:</label>
```

```
 <select id="leaveType" name="leaveType" required>
 <option value="vacation">Vacation</option>
 <option value="sick">Sick Leave</option>
 </select>
<br>
<label for="startDate">Start Date:</label>
 <input type="date" id="startDate" name="startDate" required>
 <br>
<label for="endDate">End Date:</label>
 <input type="date" id="endDate" name="endDate" required>
 <br>
<button type="submit">Submit</button>
 </form>
</body>
<style>
    body {
    font-family: Arial, sans-serif;
    text-align: center;
   margin: 50px;
   }
   h1 {
    color: #333;
   }
   form {
    margin: 20px auto;
    width: 300px;
   border: 1px solid #333;
    padding: 20px;
   }
   label {
    display: block;
   margin-bottom: 5px;
   }
   input[type="text"],
   input[type="date"],
   select {
    width: 100%;
   padding: 5px;
   margin-bottom: 10px;
   }
   button {
    background-color: #333;
   color: #fff;
   border: none;
   padding: 10px 20px;
```

```
    cursor: pointer;
   }
</style>
</html>
```

Server.js

```
const express = require('express');
const bodyParser = require('body-parser');
const mongoose = require('mongoose');
const app = express();

app.set('view engine', 'ejs'); // Uncomment this line to set EJS as the view
engine

app.use(express.static('public'));
app.use(bodyParser.urlencoded({ extended: false }));

mongoose.connect('mongodb://127.0.0.1:27017/leave_management_system', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

mongoose.connection.on('open', () => {
  console.log('Connected to MongoDB');
});

const leaveRequestSchema = new mongoose.Schema({
  employeeName: String,
  leaveType: String,
  startDate: Date,
  endDate: Date,
});

const LeaveRequest = mongoose.model('LeaveRequest', leaveRequestSchema);

app.get('/', (req, res) => {
  LeaveRequest.find({})
    .exec()
    .then(leaveRequests => {
      res.render('index', { leaveRequests });
    })
    .catch(err => {
      console.error(err);
      res.status(500).send('Internal Server Error');
```

```
    });
});

app.post('/apply', (req, res) => {
  const newLeaveRequest = new LeaveRequest({
    employeeName: req.body.employeeName,
    leaveType: req.body.leaveType,
    startDate: req.body.startDate,
    endDate: req.body.endDate,
  });

  newLeaveRequest.save();
  res.end('Submitted');
});

const port = process.env.PORT || 4000;
app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

7.

App.js

```
import React, { useState } from 'react';
import './App.css';
import TaskForm from './Components/TaskForm';
import TaskList from './Components/TaskList';
const App = () => {
const [tasks, setTasks] = useState([]);
const addTask = (description, project) => {
setTasks([...tasks, { description, project, status: 'Pending' }]);
};
const changeStatus = (index) => {
const newTasks = [...tasks];
const task = newTasks[index];
switch (task.status) {
case 'Pending':
task.status = 'InProgress';
break;
case 'InProgress':
task.status = 'Completed';
break;
case 'Completed':
```

```
task.status = 'Pending';
break;
default:
break;
}
setTasks(newTasks);
};
return (
<div className="App">
<h1>Task Management System</h1>
<TaskForm onAddTask={addTask} />
<TaskList tasks={tasks} onStatusChange={changeStatus} />
</div>
);
};
export default App;
```

App.css

```
.App {
  text-align: center;
  margin: 20px;
  }
h1 {
  color: #007bff;
  }
form {
  margin-bottom: 20px;
  }
button {
  margin-top: 10px;
  }
ul {
  list-style-type: none;
  padding: 0;
  }
li {
  margin: 5px;
  display: flex;
  justify-content: space-between;
  background-color: #f0f0f0;
  padding: 5px;
  cursor: pointer;
  }
```

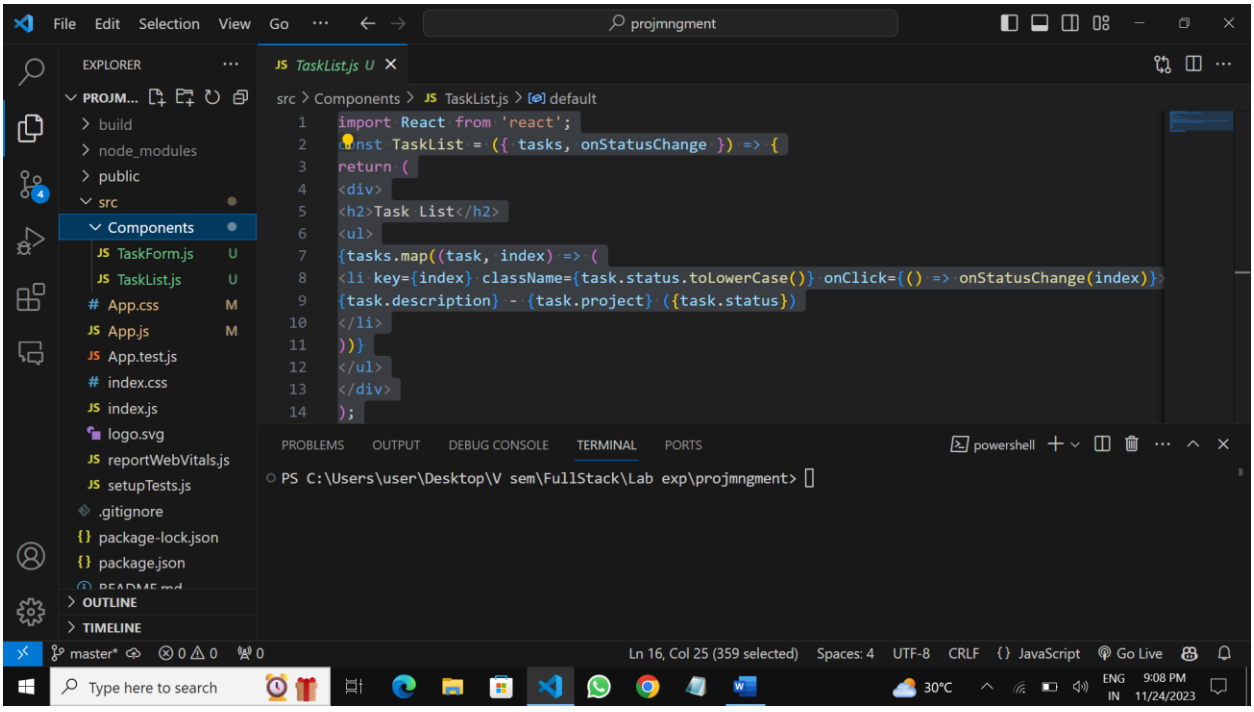
```
.completed {
  background-color: #b3ffb3;
  }
.in-progress {
  background-color: #ffffb3;
  }
```

Components/TaskForm.js

```
import React, { useState } from 'react';
const TaskForm = ({ onAddTask }) => {
const [task, setTask] = useState('');
const [project, setProject] = useState('');
const handleSubmit = (e) => {
e.preventDefault();
if (task.trim() !== '' && project.trim() !== '') {
onAddTask(task, project);
setTask('');
setProject('');

}
};
return (
<div>
<h2>Add New Task</h2>
<form onSubmit={handleSubmit}>
<input
type="text" value={task}
onChange={(e) => setTask(e.target.value)}
placeholder="Enter new task..."
/>
<input
type="text" value={project}
onChange={(e) => setProject(e.target.value)}
placeholder="Enter project name..."
/>
<button type="submit">Add Task</button>
</form>
</div>
);
};
export default TaskForm;
```

Components/TaskList.js

```
import React from 'react';
const TaskList = ({ tasks, onStatusChange }) => {
return (
<div>
<h2>Task List</h2>
<ul>
{tasks.map((task, index) => (
<li key={index} className={task.status.toLowerCase()} onClick={() =>
onStatusChange(index)}>
{task.description} - {task.project} ({task.status})
</li>
))}
</ul>
</div>
);
};
export default TaskList;
```



8.

App.js

```
import React, { useState } from 'react';
import './App.css';
```

```
import Question from './Components/Question';
const App = () => {
const [answers, setAnswers] = useState([]);
const [questions] = useState([
{
id: 1, text: 'What is your favorite color?', options: ['Red', 'Green', 'Blue',
'Yellow', 'Other'], }, {
id: 2, text: 'Which programming language do you prefer?', options: ['JavaScript',
'Python', 'Java', 'C++', 'Other'], }, {
id: 3, text: 'What is your favorite animal?', options: ['Dog', 'Cat', 'Elephant',
'Dolphin', 'Other'], }, {
id: 4, text: 'How do you like to spend your weekends?', options: ['Reading',
'Watching Movies', 'Outdoor Activities', 'Gaming', 'Other'], }, {
id: 5, text: 'What type of music do you enjoy?', options: ['Pop', 'Rock',
'Classical', 'Hip Hop', 'Other'], }, ]);
const handleAnswer = (questionId, option) => {
const updatedAnswers = answers.filter((answer) => answer.questionId !==
questionId);
setAnswers([...updatedAnswers, { questionId, option }]);
};
return (

<div className="App">
<h1>Online Survey Application</h1>
<div>
{questions.map((question) => (
<Question key={question.id} question={question} onAnswer={handleAnswer} />
))}
</div>
<h2>Your answers:</h2>
<ul>
{answers.map((answer, index) => (
<li key={index}>
Question {answer.questionId}: {answer.option}
</li>
))}
</ul>
</div>
);
};
export default App;
```

App.css

```
.App {
  text-align: center;
  margin: 20px;
  }
  h1 {
  color: #007bff;
  }
  ul {
  list-style-type: none;
  padding: 0;
  }
  li {
  margin: 5px;
  }
  label {
  margin-right: 5px;
  }
```

Components/Questions.js

```
import React from 'react';
const Question = ({ question, onAnswer }) => {
return (
<div>
<h3>{question.text}</h3>
<div>
{question.options.map((option, index) => (
<div key={index}>
<input
type="radio"
id={option}
name={question.id}
value={option}
onChange={() => onAnswer(question.id, option)}
/>
<label htmlFor={option}>{option}</label>
</div>
))}
</div>
</div>
);
```

```
};
export default Question;
```