E-commerce Project

Important Note: Please write comments in your codes and a thought note in a separate sheet while submitting the assignment. If comments and thought note is not present in your submission, marks will get deducted.

Instructions for building an e-commerce project using React, Node.js, and MongoDB. Here's a more detailed breakdown of the steps:

The project we are trying to develop is the creation of a full-featured e-commerce application using react and node. This includes the ability to list products, add product images, render and style products, add page routing, create a Node.JS server, fetch products from the backend, manage state using the reducer hook, add the bootstrap UI framework, create product and rating components, create a product details screen, create a loading and message component, use a React context to add items to the cart, create a cart screen, create a sign-in screen, connect to a MongoDB database, seed sample products and users, create sign-in and sign-up backend APIs, implement a shipping screen, select a payment method screen, create a place order screen, create an order screen, pay for orders using PayPal, display order history, create a profile screen, and publish the application to Heroku. Additionally, we want to be able to add a sidebar and search box, create a search screen, reate an admin menu, create a dashboard screen, and manage products. This will require the use of react-router-dom, axios, bootstrap, jsonwebtoken, mongoose, react-paypal-js.

Features:

1. Add page routing:[2 points] To add page routing to your project, you'll need to install the react-router-dom library and create routes for the home screen and the product screen.

2. Create a Node.js server:[1 points] To create a Node.js server for your project, you'll need to run npm init in the root folder and update the package.json file to set the type field to module. You'll also need to install the express library and create a server.js file.

3. Fetch products from the backend:[2 points] To fetch products from the backend, you'll need to set the proxy field in the package.json file and install the axios library. You can then use the state and effect hooks in your React components to retrieve and display the products.

4. Manage state with the reducer hook:[2 points] To manage state with the reducer hook, you'll need to define a reducer function and use the useReducer hook in your React components to update the state.

5. Add the Bootstrap UI framework:[1 points] To add the Bootstrap UI framework to your project, you'll need to install the react bootstrap and bootstrap libraries and update your App.js file to include the Bootstrap styles.

6. Create product and rating components:[2 points] You'll need to create separate components for

displaying products and ratings. The product component should include the rating component.

7. Create a product details screen:[2 points] To create a product details screen, you'll need to fetch the product data from the backend and display it in three columns: one for the image, one for the product information, and one for the action buttons (e.g., add to cart).

8. Create loading and message components:[2 points] You'll need to create separate components for displaying loading and error messages. The loading component should include a spinner, and the message component should include a function for displaying error messages.

9. Create a React context for adding items to the cart:[2 points] To create a React context for adding items to the cart, you'll need to define a context object and a reducer function. You can then use the StoreProvider component to wrap your application and the useContext hook to access the context in your components.

10. Complete the add to cart feature:[2 points] To complete the add to cart feature, you'll need to check if the item already exists in the cart and if there is sufficient stock in the backend.

11. Create a cart screen: [2 points]To create a cart screen, you'll need to display the items in the cart in two columns: one for the item details and one for the action buttons (e.g., increase/decrease quantity, remove item).

12. Complete the cart screen:[2 points] To complete the cart screen, you'll need to add click handlers for the action buttons (e.g., increase/decrease quantity, remove item, checkout).

13. Create a sign-in screen: [2 points]To create a sign-in screen, you'll need to create a form with email and password fields and a sign-in button.

14. Connect to a MongoDB database:[1 points] To connect to a MongoDB database, you'll need to create a MongoDB Atlas account and a local MongoDB database. You can then install the mongoose library and use it to connect to the database in your Node.js server.

15. Seed sample products: [2 points]To seed sample products, you'll need to create a Product model and a seed route in your Node.js server. You can then use the route to seed the sample products into your MongoDB database.

16. Seed sample users: [2 points]To seed sample users, you'll need to create a User model and a seed route in your Node.js server. You can then use the route to seed the sample users into your MongoDB database.

17. Create a sign-in backend API: [3 points]To create a sign-in backend API, you'll need to install the jsonwebtoken library and define a generateToken function. You can then create an API route that handles sign-in requests and returns a token.

18. Complete the sign-in screen:[3 points] To complete the sign-in screen, you'll need to handle the submit action by sending a sign-in request to the backend API and saving the token in the store and local storage. You'll also need to show the user's name in the header.

19. Create a shipping screen:[2 points] To create a shipping screen, you'll need to create form inputs for the shipping address and handle the save shipping address action. You may also want to add a checkout wizard bar to the screen.

20. Create a sign-up screen: [2 points]To create a sign-up screen, you'll need to create input forms for the user's name, email, and password. You'll also need to handle the submit action and create a backend API for creating new users.

21. Implement a select payment method screen: [3 points]To implement a select payment method screen, you'll need to create input forms for the payment method and handle the submit action.

22. Create a place order screen:[3 points] To create a place order screen, you'll need to show the cart items, payment method, and shipping address on the screen. You'll also need to calculate the order summary and display it to the user.

23. Implement the place order action:[4 points] To implement the place order action, you'll need to handle the place order action and create an API for creating new orders in the backend.

24. Create an order screen:[3 points] To create an order screen, you'll need to create a backend API for retrieving a specific order by its ID. You can then use the API in the frontend to fetch the order data and display it in two columns: one for the order information and one for the items.

25. Pay order by PayPal:[4 points] To pay an order using PayPal, you'll need to generate a PayPal client ID, create an API for returning the client ID, and install the react-paypal-js library. You can then use the PayPalScriptProvider component in the index.js file and the usePayPalScriptReducer hook in the order screen to load the PayPal script and render the PayPal button. You'll also need to implement the onApprove payment function and create a pay order API in the backend.

26. Display order history:[4 points] To display the order history, you'll need to create an order history screen and a backend API for retrieving the user's orders. You can then use the API in the frontend to fetch the order data and display it in the screen.

27. Create a profile screen:[2 points] To create a profile screen, you'll need to retrieve the user's information from the context and display it on the screen. You'll also need to create a backend API for updating the user's information and implement the API in the frontend to allow the user to update their profile.

28. Publish to Netlify:[4 points] To publish your project to Netlify, you'll need to create a new Node.js project and configure it to serve the build folder in the frontend folder. You'll

29. More Features: (Extra Features:) [5 points] then need to create a Netlify account, connect it to your GitHub repository, and create a MongoDB Atlas database. Add a sidebar and search box:[2 points] To add a sidebar and search box to your project, you'll need to create the UI for the sidebar and search box and add them to your layout.

30. Create a search screen: [3 points]To create a search screen, you'll need to show filters for refining the search results and create a backend API for searching products. You'll also need to display the search results in the screen.

31. Create an admin menu:[1 points] To create an admin menu, you'll need to define protected and admin route components and add a menu for the admin in the header.

32. Create a dashboard screen: [3 points]To create a dashboard screen, you'll need to create a dashboard UI and implement a backend API for retrieving the dashboard data. You'll also need to connect the UI to the backend to display the data.

33. Manage products:[2 points] To manage products, you'll need to create a products list UI and implement a backend API for retrieving and updating products. You'll also need to fetch the product data and display it in the UI.

34. Create a product:[2 points] To create a product, you'll need to create a create product button and implement a backend API for creating products. You'll also need to handle the button click action to open the create product form.

35. Create an edit product:[2 points] To create an edit product feature, you'll need to create an edit button and an edit product UI. You'll also need to display the product information in the input fields and implement a backend API for updating products.

36. Manage orders: [4 points]To manage orders, you'll need to create an orders list UI and implement a backend API for retrieving and updating orders. You'll also need to fetch the order data and display it in the UI.

37. Manage users:[2 points] To manage users, you'll need to create a users list UI and implement a backend API for retrieving and updating users. You'll also need to fetch the user data and display it in the UI.

38. Implement pagination: To implement pagination, you'll need to create a pagination UI and a backend API for paginating data. You'll also need to fetch the data for each page and update the pagination UI to reflect the current page.

More Features: (Extra Features:) [5 points]

1. Implement pagination: To implement pagination, you'll need to create a pagination UI and

a backend API for paginating data. You'll also need to fetch the data for each page and

update the pagination UI to reflect the current page.

2. Implement sorting: To implement sorting, you'll need to create a sorting UI and a backend API for sorting data. You'll also need to fetch the sorted data and update the UI to reflect the current sorting criteria.

3. Implement filtering: To implement filtering, you'll need to create a filtering UI and a backend API for filtering data. You'll also need to fetch the filtered data and update the UI to reflect the current filters.

4. Implement uploading images: To implement uploading images, you'll need to create an image upload UI and a backend API for storing images. You'll also need to handle the image upload action and send

the image data to the backend API.

5. Implement sending emails: To implement sending emails, you'll need to install a library like nodemailer and create a backend API for sending emails. You'll also need to

6. Implement password reset: To implement password reset, you'll need to create a password reset UI and a backend API for resetting passwords. You'll also need to handle the password reset action and send a password reset request to the backend API.

7. Implement email verification: To implement email verification, you'll need to create an email verification UI and a backend API for verifying emails. You'll also need to handle the email verification action and send an email verification request to the backend API.

8. Implement authorization: To implement authorization, you'll need to create an authorization system that verifies user permissions before allowing access to certain routes or actions. This can be implemented using libraries like passport or JWT.

9. Implement server-side rendering: To implement server-side rendering, you'll need to use a library like Next.js or ReactDOMServer to render your React components on the server. This can improve the performance and SEO of your application.

10. Implement unit testing: To implement unit testing, you'll need to install a testing library like Jest or Mocha and write test cases for your application. You can then run the tests to ensure that your code is working as expected.

Good Approach:

Welcome to the React and Node project guidelines! We're glad you're here and excited to help you get started with your project. Here are some tips to help you get started:

1. Start by researching the technologies you plan to use. Be sure to read up on best practices, tutorials, and other resources to help you get acquainted with the tools you'll be using.

2. Create a detailed plan for your project. Make sure to include a timeline for each step, a list of tasks, and any other important details.

3. Develop a coding style that you and your team can agree on. This will ensure consistency and help speed up development.

4. Have a communication plan in place. Make sure everyone has a way to reach each other

and stay on track with the project.

5. Test your code often to ensure it's working as expected.

6. Make use of version control, such as Git, to keep track of changes in the code.

7. Document your project and code. This will make it easier for future developers to

understand and maintain the code.

We hope these guidelines help you get started on your React and Node project. Good luck and have fun!

Example of Ecommerce Website:

There are many examples of e-commerce websites that you can reference for ideas and

inspiration. Here are a few examples:

1. Amazon: Amazon is a well-known e-commerce platform that sells a wide variety of products, from books and electronics to household goods and clothing.

2. eBay: eBay is an e-commerce platform that allows individuals and businesses to buy and sell new and used items.

3. Etsy: Etsy is an e-commerce platform that focuses on handmade, vintage, and unique items.

4. Shopify: Shopify is a popular e-commerce platform that allows businesses to create their own online stores and sell their products.

5. AliExpress: AliExpress is an e-commerce platform that offers a wide variety of products, including electronics, clothing, and home goods, at low prices.

These are just a few examples of e-commerce websites, and there are many more out there to explore.

Make sure that design is unique and should be different for everyone. Inspiration of design

Design can be inspired by a variety of things - the natural world, art, music, fashion, and much more. To get started, I suggest taking a walk and observing the sights, sounds, and textures around you. Try to take note of patterns, colors, and shapes that you find interesting. Once you've collected some ideas, you can then explore how they might be incorporated into your design. If you're feeling stuck, there are plenty of online resources to help you find new design inspiration. I hope this helps get your creative juices flowing!

Important Note: Please write comments in your codes and a thought note in a separate sheet while submitting the assignment. If comments and thought note is not present in your submission, marks will get deducted.