ONLINE BOOK STORE

Course Name: DAC

PRN	Name
200250120020	Biswanath Bose
200250120024	Devesh Prakash Singh
200250120064	Piyush Kumar Singh
200250120072	Rahul Kumar
200250120075	Rao Ritesh Muralidhar

About Project:

i. The project is mainly focused on building an e-commerce website which provides books of various genres for sale.
ii. There are mainly three kinds of users:

1. New User
   a) User without login credentials
   b) Every unregistered user is initially new user only
   c) Can browse through the site
   d) Can search for books
   e) Can see the book details
   f) Cannot place order
   g) Cannot give rating and review for book.
   h) Can sign-up by providing their details like name, e-mail ID, password by doing so the new user will become the registered user.

2. Registered User
   a) These are the user with login credential
   b) They have access of all the functionality of new user
   c) Can place order
   d) Can give rating and review on the books
   e) Can update their profile
   f) Can see their order details which are placed by them

3. Admin User:
   a) These users are registered user with some extra functionality
   b) Can see all the registered users, can update their details, can make any registered user as admin, can delete any of the registered users.
   c) Can see all the available books, can update their details, can delete the book and can add new book
   d) Can see all the placed orders by the registered users and can mark them as delivered.

Flow of the Project:

1. The project is based on MERN stack.
2. React is used as view part and Redux is used for the state management.
3. NodeJS is used as server side technology.
4. ExpressJS framework is used for service layer and mongoose (ODM) for database MongoDB.
5. Here we consider a scenario where registered user wishes to view a particular book details.
6. When the user visits Online Book Store web application, he/she will land on homepage.
7. On the Homepage user will be able to see the available books in the store. These books are fetched from the database. In the bookAction.js an action is created by listbooks() of type BOOK_LIST_REQUEST where a GET(/api/books) request is made to server .The server routes the req to books routes from where req is transferred to getBooks() of book controller. In the getBooks() a function find() of mongoose is called on the Book Model by passing empty object ({}) as we have to fetch all the books from the database. If the request to database successful then all books are fetched from the database as Book object array. The Book array is sent as JSON object to the client and books will be displayed to homepage by using Book component.
8. User may click on the book of his choice to see the details of book. After clicking on the book user will be routed to Book Details screen by routing to 'book/{book._id}'. The routing of client is controlled in the app.js file using Route Component which is provided by library react-router-dom. In the book.json, an action is created by listBookDetails() of type BOOK_DETAILS_REQUEST making a GET(/api/books/id) request to the server. From the server the request is routed to the books routes and book routes transfers the request to the getBookById() function of the BookController. In this function, a request is made to the database by using findById(req.params.id) on book model. If book is found, then response will be sent to the client having book object.
9. Based on the type of action created, bookDetailsReducer will change the state of virtual DOM.
10. Browser will compare the previous state and present virtual DOM state as previous state and changes observed in the state will be considered and bookDetailsScreen will get rendered.

<u>A few problems faced and how we overcame them:</u>

1) After completing the whole checkout and payment process of an order, when we want to place next order, we perform the checkout process and want to proceed to next then last order details page is rendered automatically. This was basically due to after completing the payment we were not resetting the orders state and shipping address state. To solve this issue we dispatched two actions of type ORDER_DETAILS_RESET and BOOK_DETAILS_RESET to reset the Book details and order details state to its initial value.

2) As each user can see their order list. Any user visits orders list screen after list screen and log out .In the same browser when new user logs in an want see their order list, the order list page was rendering on the previous state of the previous user to solve this we dispatch actions of type MY_ORDER_LIST_RESET to initialize order list stat to its value.

3) While admin tried to create a new book, Create Book button wasn't working. It was because during the BookCreate action we made a POST request to get the sample entry for the book, but we did not send any object to the server. It is necessary to send an object while mode of request is POST, and this solved our issue.

4) When an exception occurs on the server side, then server automatically used to crash. To handle this exception we used the asynchandler. It is a simple middleware for handling exceptions inside of async express routes and passing them to your express error handlers.

Learnings from the project:

1. We followed the Agile methodology in project development
2. ASANA project management tool was used as scrum board and a schedule oriented approach was followed.
3. Layout of presentation design using Pencil tool.
4. Maintaining the version of application using Git
5. ReactJS and its use on client side and NodeJS on server side.
6. MongoDB Atlas helped in remote accessing application data to all members
7. Learnt the use of Redux for state management; JWT tokens for authorization; Mongoose ODM for document mapping
8. Use of Redux DevTools in browser.
9. Working in collaboration as a team by using tools like GitHub, Google Drive, Zoom, AnyDesk.
10. Learnt how to work as a team and how project development works in real-time remote environment