Fullstack Developer Hiring Assignment

Problem Statement:

In this assignment, you will create a web application for showcasing a list of courses and their details using React. Your task is to implement the course listing page, a course details page, and a student dashboard to display enrolled courses. Additionally, you may attempt bonus tasks marked with 🎁 for showcasing proficiency.

Frontend Requirements:

Course Listing: Create a page to showcase list of courses

1) Fetch a list of sample courses (use a dummy api or readymade backends like firebase).
2) Display the courses in a scrollable list with basic information and enable searching based on course name and instructor (e.g., course name, instructor, etc.).
3) Users should be able to click on a course to view its details.

Course Details Screen: Create a new screen that displays detailed information about a selected course. This screen should be accessible from the course listing screen.
Use the sample course model provided as a reference for the structure of course data.
Display Course Information:

Display the following course information on the details screen (as described in the previous assignment):

1) Course name
2) Instructor's name
3) Description
4) Enrollment status (e.g., 'Open', 'Closed', 'In Progress')
5) Course duration
6) Schedule
7) Location
8) Pre-requisites
9) Syllabus as an expandable item

Student Dashboard:

1) Create a user dashboard for students to display the courses they are enrolled in. Include a user-friendly interface with a list of enrolled courses.
2) Display course name, instructor name, thumbnail, due date, and a progress bar.

3) Implement a feature that allows students to mark courses as completed.

Advanced State Management:

Utilize a state management library such as Redux or MobX to manage the application's state effectively.

Backend Requirements

Create a Retrieve Course List API:

1) Create an API endpoint that returns a list of sample courses. You can use dummy data stored on the server.
2) The API response should provide basic information for each course, including course name, instructor name, and other relevant details.

Create a Retrieve API for Course Details:

1) Create an API endpoint that returns a details of a particular course.
2) The API endpoint should provide basic and relevant infromation for the particular course mentioned above.

API for Enroll in a Course:

1) Create an API endpoint that allows students to enroll in a course. The API should accept the student's user ID and the course identifier as parameters.
2) Ensure that a student cannot enroll in the same course multiple times.

Retrieve Enrolled Courses:

1) Implement an API endpoint that returns a list of courses in which a student is enrolled.
2) The response should include course name, instructor name, a course thumbnail, due date, and a progress status for each enrolled course.

Mark Courses as Completed:

1) Create an API endpoint that allows students to mark a course as completed. The API should accept the student's user ID and the course identifier.
2) Ensure that a student can only mark a course as completed if they are enrolled in it.

Course Details for Dashboard:

1) Implement an API endpoint to provide detailed information about a specific course for the student's dashboard.
2) The API should accept the student's user ID and the course identifier as parameters.

3) Ensure that students can only access the details of courses they are enrolled in.

User Authentication:

1) Implement user authentication to ensure that only authorized students can access the dashboard, enroll in courses, and mark them as completed
2) Secure API endpoints to require authentication and authorization checks.

Pagination:

1) Implement pagination for the course list API to limit the number of courses returned per request. Include options to specify the page number and the number of courses per page.

Search Options:

1) Create an API endpoint that allows students to search for courses based on specific criteria, including but not limited to course name, instructor name, or keywords related to the course.
2) The API should accept search parameters in the request, and the search should be case-insensitive, meaning it should match courses regardless of letter case.
3) Provide optional advanced search options, such as filtering courses by enrollment status (e.g., 'Open,' 'Closed,' 'In Progress') or by course duration.
4) Allow students to customize their search to find courses that match their specific needs.

Notes:

1) Properly document the API, providing clear and concise information on how to use each endpoint.
2) Implement robust error handling to manage various scenarios, such as unauthorized access or enrolling in non-existent courses.
3) If you choose to use a database to store course and enrollment data, ensure that the database is set up and accessible for testing purposes(Use Docker to ensure this).

Submission Guidelines:

1) Please submit your assignment as a GitHub repository. Include the code for the course listing page, course details page, student dashboard, and any related components or screens. Additionally, provide clear instructions on how to run your application.

2) Attach a demo video of the website.

Evaluation Criteria:

1) Correct implementation of the course listing page, course details page, and student dashboard.
2) Proper functionality for searching, and navigating between screens.
3) Proper data fetching and display.
4) Effective use of state management with Redux or MobX.
5) Responsive design that works well on different devices.
6) Note: You don't need to provide excessive focus on User interface design and visual appeal. You can choose a design of your choice and move forward with it. We are more interested in checking the quality of code and the functionality as you'll be working with skilled UI/UX resources in the future,

🎁) Implement real time connection with api that shows the number of likes instantly in course listing page. Note that the feature to like a course is not necessary, you should be able to update data in the backend and it will show up instantly in the frontend.

Sample Course Model: A sample course model is attached for your reference. Make necessary changes if required.

```
const courseModel = {
  id: 1, // Unique identifier for the course
  name: 'Introduction to React Native',
  instructor: 'John Doe', // Name of the course instructor
  description: 'Learn the basics of React Native development and
build your first mobile app.',
  enrollmentStatus: 'Open', // Can be 'Open', 'Closed', or 'In
Progress'
  thumbnail: 'your.image.here', //Link to the course thumbnail
  duration: '8 weeks', // Duration of the course
  schedule: 'Tuesdays and Thursdays, 6:00 PM - 8:00 PM',
  location: 'Online',
  prerequisites: ['Basic JavaScript knowledge', 'Familiarity with
React'],
  syllabus: [
    {
      week: 1,
      topic: 'Introduction to React Native',
      content: 'Overview of React Native, setting up your
development environment.'
```

```
    },
    {
      week: 2,
      topic: 'Building Your First App',
      content: 'Creating a simple mobile app using React Native
components.'
    },
    // Additional weeks and topics...
  ],
  students: [
    {
      id: 101,
      name: 'Alice Johnson',
      email: 'alice@example.com',
    },
    {
      id: 102,
      name: 'Bob Smith',
      email: 'bob@example.com',
    },
    // Additional enrolled students...
  ],
};

export default courseModel;
```