Below is the assignment:

MERN Stack Developer Assignment

Here's the application flow for the MERN stack developer assignment:

1. User Registration and Authentication:

- User accesses the application and navigates to the registration page.
- User fills out the registration form with required details including email, password, profile image etc.
- The frontend displays a preview of the selected profile image before submission.
- Upon submission, the backend validates the input data, sends an email verification link to the provided email address, and stores the user data in the MySQL database using Sequelize.
- User receives the email verification link and clicks on it to verify their email address.
- Once the email is verified, the user can log in to the application using their email and password.
- The backend verifies the user's credentials, generates a JWT token with a refresh token, and sends it back to the client.

2. Password Reset:

- If a user forgets their password, they can request a password reset.
- User provides their email address and requests a password reset.
- Backend verifies the email address and sends a password reset link to the user's email.
- User clicks on the password reset link and is directed to a page where they can reset their password.
- After entering a new password, the backend updates the user's password in the database.
- Show the list of users so that he can view any user profile, search any user.
- Create a my profile page with edit profile functionality.

3. Social Login with Google(optional):

- User chooses to log in with Google.
- User is redirected to Google's authentication page where they log in with their Google credentials.
- Upon successful authentication, Google redirects the user back to the application with an authentication token.
- The backend verifies the authentication token with Google and creates a new user account or logs in the existing user based on the Google account information.

4. Role-Based Access Control (RBAC):

- Upon successful authentication, the backend checks the user's role and permissions stored in the database.
- Based on the user's role, certain features or functionalities may be restricted or accessible.
- For example, an admin user may have access to certain administrative features while a regular user may have access to standard functionalities, like if logged in by admin then show the admin actions like deleting any user.

6. Pagination and Search:
- User navigates to a page where a list of items is displayed.
- The frontend implements pagination to limit the number of items displayed per page.
- User can navigate through different pages to view additional items.
- User can also use a search feature to filter items based on specific criteria.

8. React-Hook-Form for Form Validation:
- User fills out a form on the frontend.
- React-Hook-Form library validates the form inputs in real-time, providing instant feedback to the user.
- Invalid inputs are highlighted, and error messages are displayed to guide the user on how to correct them.

9. Responsive Design:
- The application is designed to be responsive and compatible with various devices and screen sizes.
- User can access and use the application seamlessly across desktop, tablet, and mobile devices.

This flow provides a general overview of how users interact with the application and the key features and functionalities implemented in both the backend and frontend.

<u>Here is the technical requirements for the development:</u>

Backend Requirements:

1. Implement a MySQL database using Sequelize ORM for data storage.
2. Integrate Redis for caching to improve application performance.
3. Implement JWT authentication with refresh token feature for user authentication and authorization.
4. Utilize Multer for handling file uploads.
5. Implement express-validation for validating incoming requests.

6. Implement lazy-loading to optimize the loading of resources.
7. Integrate social login with Google for user authentication.
8. Implement Role-Based Access Control to manage user permissions.
9. Implement Email Verification and Password Reset functionality.

Frontend Requirements:

1. Develop a React.js frontend application.
2. Implement pagination for better navigation through data.
3. Implement a search feature to allow users to search for specific content.
4. Develop an image upload field with a preview feature.
5. Utilize the react-hook-form library for form validation.
6. Use Axios for making API requests to the backend.
7. Ensure a responsive design that looks good on various devices and screen sizes.

Submission Guidelines:

1. Create a repository (provide the link to your repository).
2. Develop the application following the provided requirements.
3. Once you have completed the assignment, submit the link to your repository.

Note: Feel free to use any additional libraries or tools that you find necessary to accomplish the tasks.

Additional Information:

- If you encounter any issues or have questions regarding the assignment, feel free to reach out for clarification.
- We value clean, well-structured, and maintainable code. Focus on writing high-quality code that adheres to best practices.
- Good luck, and we look forward to reviewing your submission!

<u>Deadline: 3 days</u>