Coding Challenge for Custom Form Builder

Overview:

Create a web application that enables organizations to design custom forms tailored to their specific requirements. This application should allow organizations to create a variety of form fields and layouts, with users being able to fill out these forms online.

Technical Requirements

General

- The application must feature user authentication, allowing organizations to register, log in, and manage their accounts.
- Organizations should be able to create, edit, and delete custom forms.
- The application should support various form field types, including text inputs, dropdown menus, checkboxes, and radio buttons.
- Forms must be dynamically rendered based on the organization's design specifications.
- All form data should be securely stored in a MySQL database, with each submission linked to its respective organization.

Backend (Express.js)

- Initialize an Express server to manage API requests.
- Implement routes for user authentication (signup, login, logout) and form management (creation, editing, deletion).
- Integrate the server with a MySQL database using the mysql2 package.
- Secure user authentication using JWT or session-based methods. Ensure passwords are securely hashed.
- Develop endpoints for form management, including creation, updating, and deletion.
- Implement dynamic form rendering based on database-stored form designs.
- Handle form submissions securely, storing data in the MySQL database.

Frontend (React)

- Initialize a new React project for the application's frontend.
- Create components for user authentication (signup, login, logout).
- Implement robust form validation and error handling.
- Develop an interface for organizations to manage their forms.
- Design components for creating and editing form fields and forms.
- Dynamically render forms for user submission based on backend specifications.

Database (MySQL)

- Design and implement tables for users, organizations, forms, and form fields.
- Establish relationships between tables to reflect the organizational structure and form hierarchy.

Security and Validation

- Implement frontend and backend validation to ensure data integrity.
- Sanitize all inputs to prevent SQL injection and other common web vulnerabilities.
- Employ CSRF protection measures.

Documentation

- Provide detailed documentation on setting up and running the application, including environment setup and server start instructions.
- Document the API endpoints and their usage.

Testing

- Write unit tests for critical components of the application.
- Perform integration testing to ensure seamless frontend and backend interaction.

CI/CD

- Implement Continuous Integration and Continuous Deployment using GitHub Actions.
- Ensure automated tests are part of the CI/CD pipeline.
- Document the CI/CD process, including setup and usage instructions.

Submission Guidelines

- Repository Setup: Create a private GitHub repository for your project. Develop your solution on a new branch within this repository.
- README: Include a README file with detailed instructions on how to set up and run your application locally.
- Collaborator Invitation: Upon completion, invite hr@technimus.com (HmTechnimus) as collaborators to your private repository.
- Keep It Private: Ensure your repository remains private. Public repositories will be considered invalid submissions.

Evaluation Criteria

- Functionality: Does the application meet the outlined requirements?
- Code Quality: Is the code well-organized, readable, and maintainable?
- User Experience: How intuitive and user-friendly is the application?
- Security: Has the application implemented measures against common security threats?
- Documentation: Is the project documentation clear and helpful for setting up and understanding the application?

Good luck with your challenge! We look forward to seeing your innovative solutions.