**Assignment: React useForm Implementation for E-commerce Application**

**Objective**

Develop two forms for an e-commerce application using the React **useForm** hook from the React Hook Form library. The first form is for user registration and the second one for the checkout process. Both forms should have appropriate fields and validations.

**Tasks**

**1. User Registration Form**

Create a user registration form with the following fields and validations:

* **Full Name**: Required, only alphabetic characters
* **Email Address**: Required, valid email format
* **Password**: Required, minimum 8 characters, including at least one number, one uppercase, and one lowercase letter
* **Confirm Password**: Required, must match the password

**2. Checkout Form**

Create a checkout form with the following fields and validations:

* **Shipping Address**: Required
* **City**: Required, only alphabetic characters
* **Postal Code**: Required, numeric, format specific to the country
* **Country**: Required, selection from a dropdown
* **Payment Method**: Required, selection from a dropdown (options like Credit Card, PayPal, etc.)

**Implementation Requirements**

* Use the **useForm** hook from React Hook Form for form handling.
* Implement form submission handling that logs form data to the console.
* Include validation feedback to the user for each field (e.g., error messages when validations fail).
* Ensure the form is responsive and user-friendly.

**Bonus (Optional)**

* Add custom styling to the forms.
* Implement more complex validations (e.g., asynchronous validation for email uniqueness).
* Create a dynamic dropdown for the country field, populated from an external source.

**Deliverables**

Submit a GitHub repository containing:

* Source code for both forms.
* A README file with instructions on how to run and test the application.

**Evaluation Criteria**

* Correct implementation of **useForm** and form validations.
* Code readability and structure.
* Proper handling of form submission.
* Functionality of the forms according to the requirements.

This assignment provides a comprehensive task for creating functional forms in a React application, using the **useForm** hook for form handling and validations. It covers fundamental aspects of form handling in React and can be a good exercise to understand and apply the concepts of React Hook Form.

Here's a pseudocode for implementing the User Registration and Checkout forms using React's **useForm** hook. This pseudocode provides an abstract representation of what your React components and logic might look like.

**User Registration Form Pseudocode**

Component: RegistrationForm

- Initialize useForm hook to manage form data, validations, and submissions

Function: onSubmit

- Input: formData

- Action: Log formData to the console

Render:

- Form Element with onSubmit event handler

- Input for Full Name with:

- Required validation

- Alphabetic characters validation

- Display error message if validation fails

- Input for Email Address with:

- Required validation

- Email format validation

- Display error message if validation fails

- Input for Password with:

- Required validation

- Minimum 8 characters, including a mix of upper and lower case letters and numbers

- Display error message if validation fails

- Input for Confirm Password with:

- Required validation

- Match with Password field

- Display error message if validation fails

- Submit button

**Checkout Form Pseudocode**

Component: CheckoutForm

- Initialize useForm hook to manage form data, validations, and submissions

Function: onSubmit

- Input: formData

- Action: Log formData to the console

Render:

- Form Element with onSubmit event handler

- Input for Shipping Address with:

- Required validation

- Display error message if validation fails

- Input for City with:

- Required validation

- Alphabetic characters validation

- Display error message if validation fails

- Input for Postal Code with:

- Required validation

- Numeric format validation

- Display error message if validation fails

- Dropdown for Country with:

- Required validation

- Options populated from an external source

- Display error message if validation fails

- Dropdown for Payment Method with:

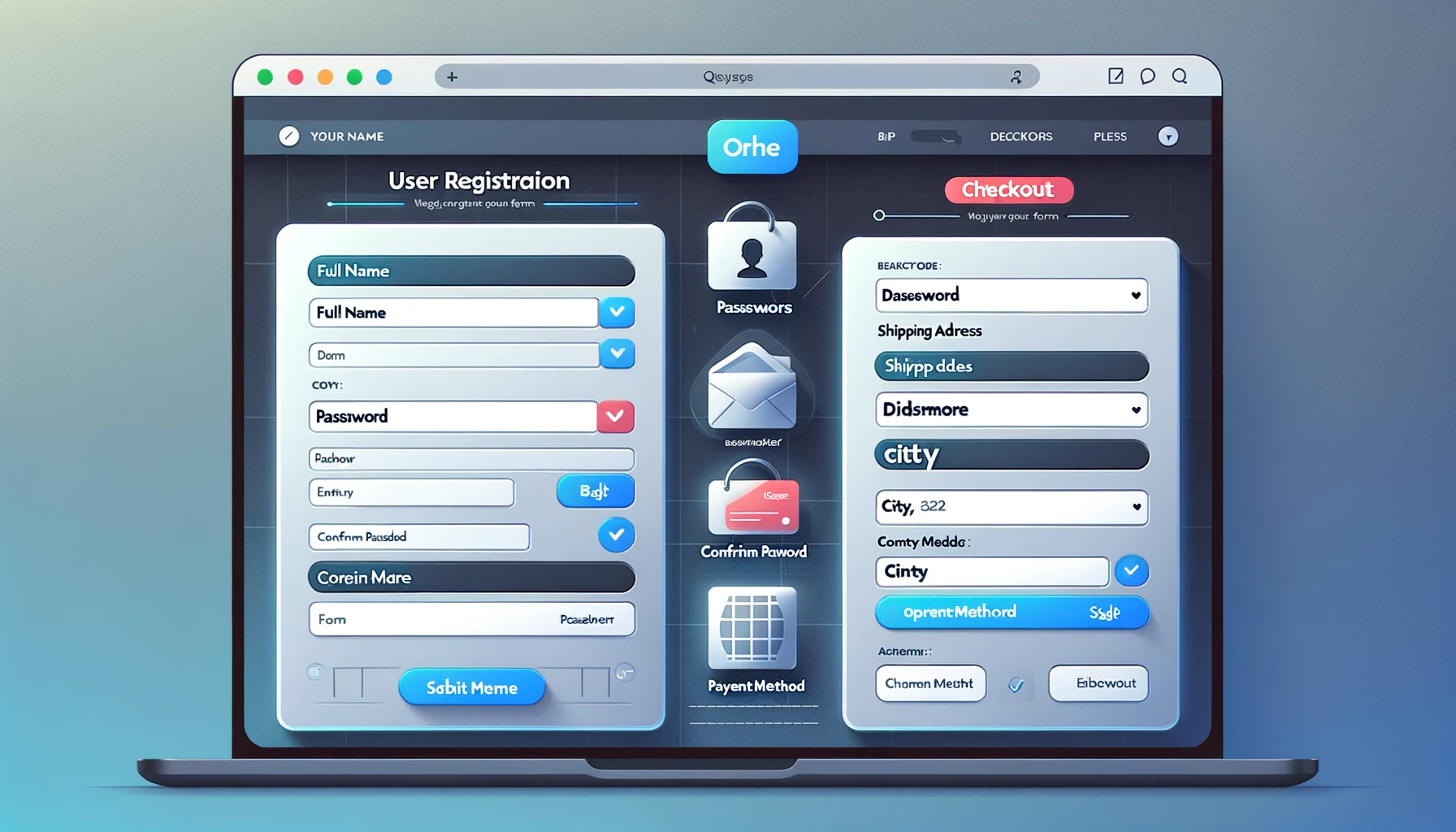
- Required validation

- Options like Credit Card, PayPal, etc.

- Display error message if validation fails

- Submit button

**Sample output**



Here is the sample output image depicting a web application interface with two forms. On the left, you can see the 'User Registration Form,' and on the right, the 'Checkout Form,' each with the specified fields and error messages. This visual representation showcases how these forms might appear in a modern, clean style suitable for an e-commerce website.