**React Hook Form**

1. Introduction
   1. Drag & drop headless CMS
   2. One Stop Shop For React
      1. React fundametals and advanced topics
      2. React hooks
      3. React typescript
      4. React redux
      5. React router
      6. React testing
      7. React material UI
      8. React styled components
      9. React storybook
      10. React query
      11. React table
      12. React formik
      13. React render
      14. React Hook Form
   3. Forms
      1. Crucial aspect of any business application
      2. Users
         1. Registering
         2. Logging in
         3. Making purchases
         4. Scheduling appointments
      3. Developers
         1. Hadling form
         2. Enforcing validation
         3. Providing visual feedback
         4. Data submission
   4. React Hook From
      1. What?
         1. A small library that helps deal with forms in React.
      2. Why?
         1. Manage form data.
         2. Submit form data.
         3. Enforce validations.
         4. Provide visual feedback.
         5. React Hook Form provides a simple, scalable, and performant way to manage even the most complex of forms.
      3. How?
         1. Precisely what we will be learning in this series
   5. Prerequissites
      1. HTML
      2. CSS
      3. JavaScript
      4. React Fundamentals
2. Form Setup
   1. Project Setup
      1. Vite: npm create vite@latest rhf-demo
         1. For framework, select React
         2. For variant, choose TypeScript

cd rhf-demo

npm install

npm run dev

* + 1. CRA
       1. npx create-react-app rhf-demo –template typescript

cd rhf-demo

npm start

1. useForm Hook
   1. Settings

npm install react-hook-form

* 1. React Hook Form
     1. Manage form data
     2. Submit form data
     3. Enforce validations and provide visual feedback

1. Managing Form State

Every form has a few moving parts that keep changing from the time a user loads the form to the time they submit it.

form state:

* current value of every field in the form
* whether a field has been interacted with
* whether a field’s value has changed
* whether the form is invalid
* whether a field contains erros

Managing Form State contd. (Trong code, để biểu thị Form State).

{

values: {…}

visited: {…}

errors: {…}

isValid: boolean

}

1. DevTools Visualization.

Settings: npm install -D @hookform/devtools

1. Form State and Rerenders
2. Form Submission
3. Form Validation
   1. Progress check
      1. Manage form data
      2. Submit form data
      3. Enforce validations and provide visual feedback
   2. Form Validation

React Hook Form supports various HTML validation rules

* + Required
  + minLength & maxLength
  + min & max
  + pattern

1. Display Error Messages
2. Custom Validation
3. Enhancing React Hook Form

* Three pillars of dealing with forms
  + Form state management
  + Form submission
  + Form validation
* Additional Features (Phức tạp hơn)
  + Form state
  + Form submission
  + Form validation

1. Default Values
2. Nested Objects
3. Arrays
4. Dynamic Fields
5. Numeric and Date Values
6. Watch Field Values
7. Get Field Values
8. Set Field Value

options:

* shouldValidate
* shouldDirty
* shouldTouch

1. Touched and Dirty States
2. Disabling Fields
3. Handle Submission Error
4. Disable Form Submission
5. Form Submission State

Useful for tracking the progress and outcome of form submissions

* isSubmitting
* isSubmitted
* isSubmitSuccessful
* submitCount

1. Reset Form
2. Async Validation
3. Validation Modes

Options:

* onSubmit
* onBlur
* onChange
* all

1. Manually Trigger Validations

* Thêm trigger ở form
* Nếu muốn tất cả thì   
                  <button

                    type='button'

                    onClick={() => trigger()}>

                    Validate

                </button>

* Nếu muốn chỉ 1 giá trị thì

                <button

                    type='button'

                    onClick={() => trigger(“channel”)}>

                    Validate

                </button>

* Section Summary
  + Default values
  + Object and array values
  + Dynamic fields
  + Numeric and date values
  + Watch, get and set field values
  + Touched and dirty states
  + Handle submission errors
  + Form submission state
  + Reset form
  + Async validation
  + Validation modes
  + Manually trigger validations

1. Yup Integration

Settings: npm i yup @hookform/resolvers

1. Zod Integration

Settings: npm i zod @hookform/resolvers

1. Material UI Integration

Settings: npm install @mui/material @emotion/react @emotion/styled