

Assignment 5: Forms Controlled Inputs

Awareness, Application, Mastery, Influence

Gaurab Wagle

June 19, 2025

Overview of the Assignment

This assignment mainly focuses on the form handling where we will have to manage form state, handle user input, and implement form submission workflows. After the completion, it has potential to build a strong foundation on the controlled components.

GitHub Repository Link

Get the complete code and project details on my GitHub repository:

<https://github.com/gxaurab/Tangible/tree/main/Assignment/src/components/Part5>

Deployed Site:

<https://tangible-production.up.railway.app/ass5/awareness>

1 Awareness

- **Rubric:** Use `useState` to control simple input fields and reflect their values in state.
- **Proof of Work:** Build a basic login form with controlled inputs for email and password fields, showing entered values below the inputs.

1.1 Approach

This part of the assignment required me to update and display the values typed in the input field dynamically. I applied the conditional rendering and just some updates on the state value. This was a simple fun assignment.

1.2 Outcome



Figure 1: Form showing the dynamic rendering of the input values

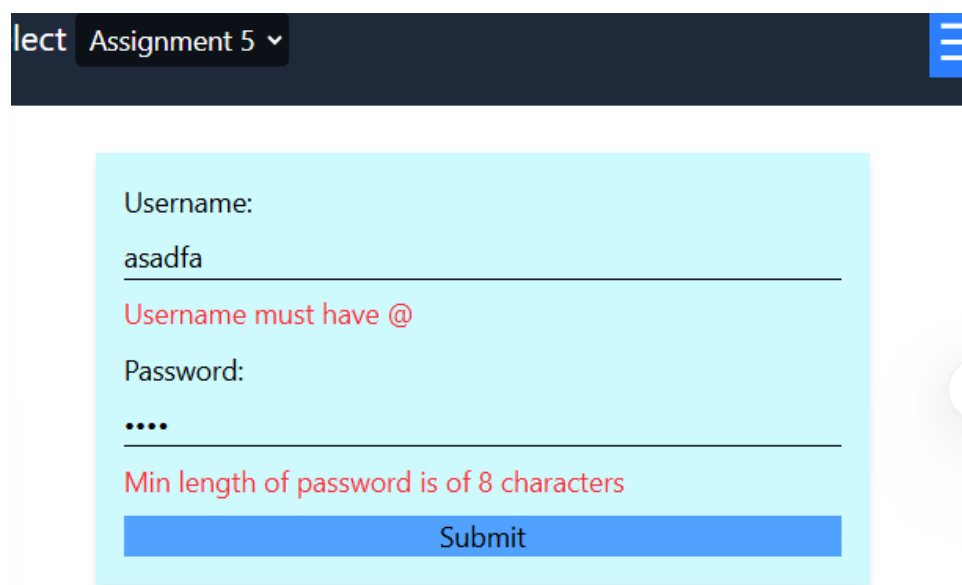
2 Application

- **Rubric:** Handle form submission events and validate required fields before allowing submit.
- **Proof of Work:** Add validation to the login form to show error messages if email or password fields are empty upon submission.

2.1 Approach:

For this assignment, I explored the react-hook-form named library to implement the validation. I just included the two fields, Username and Password. When the user submits the form, it then renders the error encountered. I found this assignment fruitful because i learned about usage of react-hook-form which I have later used in the Mastery assignment in a better way.

2.2 Outcome



The screenshot shows a dark blue header bar with the text "lect Assignment 5" and a dropdown arrow. Below the header is a light blue form container. Inside the form, there are two input fields. The first field is labeled "Username:" and contains the text "asadfa". Below this field is a red error message: "Username must have @". The second field is labeled "Password:" and contains four dots. Below this field is a red error message: "Min length of password is of 8 characters". At the bottom of the form is a blue "Submit" button.

Figure 2: Form with Error Handling using React-Hook-Form

3 Mastery

- **Rubric:** Manage multiple form fields together and handle more complex validations.
- **Proof of Work:** Build a contact form with fields like name, email, message, and validate each input with appropriate rules.

3.1 Approach

This assignment was an upgrade to the "Application", as it required me to handle the multiple form fields and required me to include some other validations, some extra validations. So I included extra field of Name, Email, Message.

Fun thing: *"I didn't know about the touched state until this assignment"* But I wondered, how could I show the errors in typing time. I was searching in the react-hook-form library but couldn't find it.

3.2 Outcome

select Assignment 5

Welcome to the Validation logic implemented in this form

Name:

aaff

Email

sa@

Enter a valid email address

Message

asfafa@

Message should be minimum of 10 character

Send

What you submitted

Figure 3: Different form fields showing the Validation Logic

4 Influence

- **Rubric:** Create reusable form input components and custom hooks to simplify validation and state.
- **Proof of Work:** Build a useFormField hook to manage input value, touched state, and validation, then reuse it for multiple inputs.

4.1 Approach

Here comes the main assignment, which took me nearly 2 days to figure out and complete.

This assignment required me to try use the custom validation logic, without using the external libraries. Here are the components I made and how I worked through:

- **Custom Hook - useFormField:** I made a custom hook that accepted a default value and the set of validation rules. I defined the types like minLength, required, maxLength which were optional as a email field wouldn't need a min or max length. Inside of the hook I checked the focus state, I created a validate function which had a logic to set the error messages according to the input we get from the user, like: if the required is true and value is empty(length 0), then it would show the error

message like: "This field is required". This hook handled the input values, touch state and error state.

- **Input Component:** A reusable input component was made which accepted certain props like name, placeholder, onChange and various props. The problem i encountered was that I needed the message text area to be a bit large. This component also conditionally rendered the label, despite I passed the label for all components and the error. Fun fact is, i figured out the usage of touch and I had fun doing this.

I realized how a simple logic solved my problem I had on the "Mastery" assignment, about the touch state.

- **Main Component::** Here i just assembled all the hook and the Input component. It included some other things like passing the values to the props about the different constraints in my input field, and handling the submit of the form.

4.2 Outcome

: Below is the Screenshot of the working Influence component. I didn't use any external libraries for that, and then I realized why those external libraries are this powerful and how they reduce the code writing time.

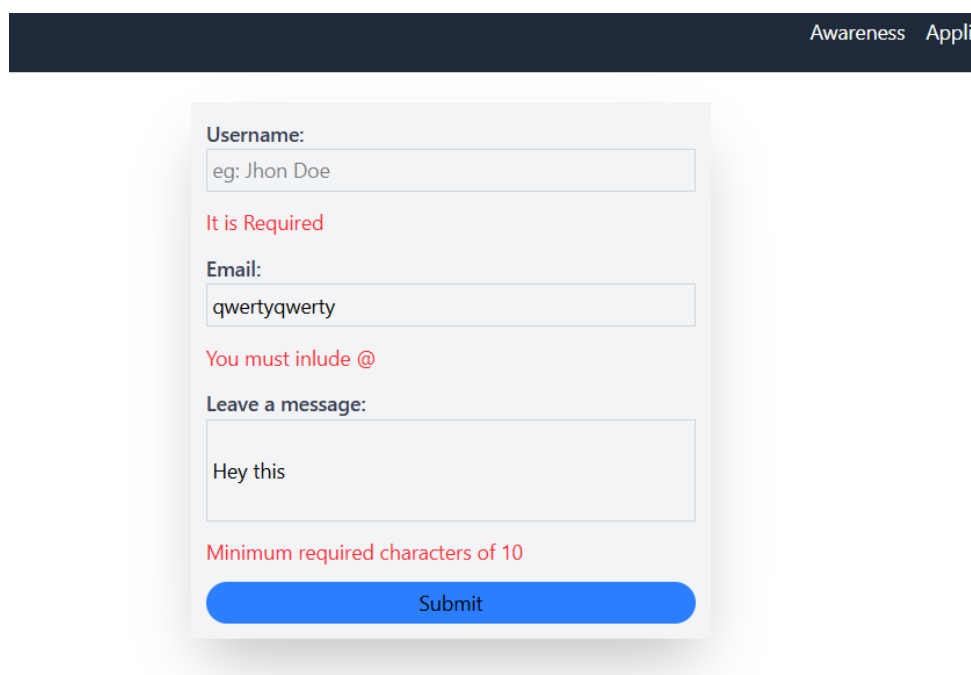
A screenshot of a web form titled "Influence Component". The form is displayed on a dark blue header bar with the text "Awareness" and "Appli" on the right. The form itself is a light gray box with a shadow. It contains four input fields: "Username:" with a placeholder "eg: Jhon Doe", "Email:" with a placeholder "qwertyqwerty", and "Leave a message:" with a placeholder "Hey this". Below the "Email:" field, there is a red error message "You must include @". Below the "Leave a message:" field, there is a red error message "Minimum required characters of 10". At the bottom of the form is a blue "Submit" button. Above the "Username:" field, there is a red error message "It is Required".

Figure 4: Influence Component

4.3 Conclusion:

As a conclusion, I am confident in form handling and I also liked the react-hook-form library. I will be using the library for my other projects. But I still have a confusion about the validation we do in the backend and frontend. Why do we require a validation in frontend if we do it in the backend?