# React Form Validation Interview Q&A

### 1. What is the purpose of useState in React?

useState is a React hook used to declare and manage state variables in functional components. In our form, we used it to store input values (like name, email, password) and track errors and submission state.

### 2. How does handleChange work in a form?

handleChange is an event handler triggered when a user types in an input. We use e.target.name to identify which input field was changed, and e.target.value to get the new value. Then we update the state using setForm({...form, [name]: value}).

### 3. Why do we use e.preventDefault() in handleSubmit?

By default, submitting a form reloads the page. In React, we use e.preventDefault() to prevent that and handle the form submission manually using JavaScript.

### 4. What is Partial<FormData> and why did we use it?

Partial<FormData> is a TypeScript utility type that makes all keys of FormData optional. We used it for our 'errors' state because not all fields will have errors at the same time.

### 5. How does validate() function help in form validation?

The validate function checks each input field for specific rules (e.g. required fields, correct email format, password length). It builds an error object and returns it. If the object is empty, we know the form is valid.

### 6. How do we conditionally render error messages?

We use conditional rendering like {errors.email && {errors.email} } to only show the message if that field has an error.

### 7. Why do we use Object.keys(errors).length === 0?

This checks if the error object is empty. If it's empty, it means there are no validation issues, so we can safely submit the form.

### 8. What is the benefit of dynamic input handling with [e.target.name]?

# React Form Validation Interview Q&A

It lets us write a single handler for all input fields. We don't need separate onChange functions for each field. It's cleaner and scalable.

### 9. How would you explain controlled vs uncontrolled inputs?

Controlled inputs are linked to React state via useState. We control their value with state. Uncontrolled inputs rely on direct DOM access using refs.

### 10. How does this demonstrate real-time validation?

While we're currently validating on submit, we could trigger validate() inside handleChange to get real-time feedback as the user types.