**React Hook Form**

**Key Features:**

- **Performant:** Reduces code and minimizes re-renders.

- **Flexible:** Easy-to-use with built-in validation support.

**Validation Rules:**

1. **`required`:** Field must not be empty.

2. **`min`:** Minimum numerical value.

3. **`max`:** Maximum numerical value.

4. **`minLength`:** Minimum number of characters.

5. **`maxLength`:** Maximum number of characters.

6. **`pattern`:** Matches a regular expression.

7. **`validate`:** Custom validation logic.

**Validation Libraries:**

**- Yup:**

- Schema-based validation.

- Supports asynchronous validation.

- Integrates well with Formik.

**- Zod:**

- Type-safe with TypeScript integration.

- Lightweight and fast.

- Supports asynchronous validation.

**Common Form Functions:**

- **`register`:** Registers fields for validation.

- **`formState`:** Provides form state info (e.g., errors).

- **`watch`:** Observes field values.

- **`handleSubmit`:** Manages form submission.

- **`reset`:** Resets the form.

- **`resetField`:** Resets a specific field.

- **`control`:** Manages form inputs, especially custom components.

- **`unregister`:** Removes fields from the form.

- **`setError`:** Sets custom errors.

- **`clearErrors`:** Clears field errors.

- **`setValue`:** Sets field values programmatically.

- **`setFocus`:** Focuses on a field.

- **`getValues`:** Retrieves current field values.

- **`getFieldState`:** Gets field state (e.g., value, errors).

- **`trigger`:** Manually triggers validation.

**Hooks:**

- **`useController`:** Access field values, errors, and methods.

- **`useFormContext`:** Access form context in nested components.

- **`useWatch`:** Watch specific field values.

- **`useFormState`:** Provides form state information.

- **`useFieldArray`:** Manage arrays of fields in a form.

Redux manages application state with:

* **Store**: Central state container.
* **Actions**: Describe events with a type and optional payload.
* **Reducers**: Update state based on actions.
* **Dispatch**: Sends actions to the store.
* **Middleware**: Handles side effects and async actions (e.g., Redux Thunk).
* **Selectors**: Extract specific data from the state.