Dynamic Forms with Configuration

- Instead of hardcoding form fields, you can define a **configuration object** that describes the form's structure, validation rules, and behavior.
- This approach allows you to dynamically render forms based on the configuration, making your code more modular and maintainable.

Advantages of Dynamic Forms

- **Flexibility**: Easily add or remove fields by updating the configuration.
- **Reusability**: The same form logic can be reused across different forms.
- **Maintainability**: Centralized configuration makes the code easier to maintain.
- **Scalability**: Suitable for complex forms with many fields and validation rules.

Steps to Create a Dynamic Form

1. **Define a Configuration Object**:
    - The configuration object describes the form fields, their types, validation rules, and other properties.
2. **Create a Custom Input Component**:
    - A reusable input component that renders different input types (e.g., text, number, select) based on the configuration.
3. **Manage Form State**:
    - Use React's `useState` or `useReducer` to manage the form's state.
4. **Handle Form Submission**:
    - Validate the form data and handle the submission.
5. **Implement Validation**:
    - Add validation logic for each field based on the configuration.