**Formik - Validation with Yup  
  
Level-1:   Without Formik  
  
 - We designed custom function for validation  
 - Custom validation messages  
 - Custom Binding of Events  
  
Level-2:  With Formik  
 - Custom Function for Validation  
 - Custom Validation Messages  
 - No Binding Events required, We use pre-defined events of formik.  
  
Level-3:  Formik with Yup  
- No custom validation  
- Pre-defined Validation Schema  
- Pre-defined Errors  
- We use pre-defined events of formik  
  
Level-4: Formik Components  
- Formik provides pre-defined components for designing a form, form field and error message.  
- From level-1 to 3 we used HTML form, span, div, input, select etc..  
- Formik provides components  
  
            <Formik>  
            <Form>  
            <Field>  
            <ErrorMessage>  
  
<Formik>            : It configures a container that can handle form.  
<Form>                : It configures a Form with set of fields.  
<Field>                : It configures a form element like text, checkbox, radio, etc..  
<ErrorMessage>    : It configures a container that can catch the errors returned  
                          by name.  
  
  
Syntax:  
  
<Formik initialValues={}  validationSchema={}  onSubmit={()=>{}}>  
    <Form>  
      {  
         <Field name="Mobile" type="text|number|email|password.."></Field>  
          <Field as="select | textarea | datalist | meter | progress">  
        <ErrorMessage name="Mobile"></ErrorMessage>  
      }  
    </Form>  
</Formik>  
  
Model Binding - Name and Location Mapping - MVC  
  
View:  
<Field>  
<ErrorMessage>  
  
Controller:  
 handleChange  
 handleBlur  
 handleSubmit  
  
Model:  
 name  
 value  
 checked  
  
Ex: YupValidationComponent  
  
  
import { useFormik, Formik, Field, ErrorMessage, Form } from "formik";  
import  \* as yup from 'yup';  
  
export default function YupValidationComponent()  
{  
    return(  
        <div className="container-fluid">  
            <h2>Register User</h2>  
            <Formik initialValues={  
                {  
                    UserName: '',  
                    Email:'',  
                    Age: '',  
                    City: ''  
                }}  
                validationSchema={  
                   yup.object({  
                     UserName: yup.string().min(4,'Name too short').max(10, 'Name too long').required('Name Required'),  
                     Email: yup.string().email('Invalid Email').required('Email Required'),  
                     Age: yup.number().required('Age Required'),  
                     City: yup.string()  
                   })  
                }  
                onSubmit={  
                    values=>{  
                        alert(JSON.stringify(values))  
                    }  
                }  
                >  
                <Form>  
                    {  
                       <div>  
                         <dl>  
                            <dt>User Name</dt>  
                            <dd><Field name="UserName" type="text"></Field></dd>  
                            <dd className="text-danger"><ErrorMessage name="UserName"></ErrorMessage></dd>  
                            <dt>Email</dt>  
                            <dd><Field name="Email" type="text"></Field></dd>  
                            <dd className="text-danger">  
                                <ErrorMessage name="Email"></ErrorMessage>  
                            </dd>  
                            <dt>Age</dt>  
                            <dd><Field name="Age" type="text"></Field></dd>  
                            <dd className="text-danger">  
                                <ErrorMessage name="Age"></ErrorMessage>  
                            </dd>  
                            <dt>City</dt>  
                            <dd><Field name="City" as="select">  
                                 <option>Delhi</option>  
                                 <option>Hyd</option>  
                                </Field></dd>  
                        </dl>  
                         <button>Register</button>  
                       </div>  
                    }  
                </Form>  
            </Formik>  
        </div>  
    )  
}  
  
                                        Validation State  
- Validation state defines the validation status of form and field.  
- It can crawl over all fields in a form and verify the validation state:  
                true        : valid  
                false        : invalid  
  
- Formik and Yup provides validation state at 2 levels  
        a) Form State Validation        [ Verifying All Fields]  
        b) Input State Validation        [ Verifying specific Field]  
  
- For Input state validation you can use <ErrorMessage>  
- For Form state validation you can use the validation properties returned by <Form> component.  
- Form Validation State includes  
  
        a) isValid                true if all fields are valid  
        b) dirty                true if any one field modified  
        c) touched            true on blur  
  
        props => <Form</Form>  
  
Ex: Validation State  
  
import { useFormik, Formik, Field, ErrorMessage, Form } from "formik";  
import  \* as yup from 'yup';  
  
export default function YupValidationComponent()  
{  
    return(  
        <div className="container-fluid">  
            <h2>Register User</h2>  
            <Formik initialValues={  
                {  
                    UserName: '',  
                    Email:'',  
                    Age: '',  
                    City: ''  
                }}  
                validationSchema={  
                   yup.object({  
                     UserName: yup.string().min(4,'Name too short').max(10, 'Name too long').required('Name Required'),  
                     Email: yup.string().email('Invalid Email').required('Email Required'),  
                     Age: yup.number().required('Age Required'),  
                     City: yup.string()  
                   })  
                }  
                onSubmit={  
                    values=>{  
                        alert(JSON.stringify(values))  
                    }  
                }  
                >  
                {  
                    fields =>  
                    <Form>  
                        {  
                       <div>  
                         <dl>  
                            <dt>User Name</dt>  
                            <dd><Field name="UserName" type="text"></Field></dd>  
                            <dd className="text-danger"><ErrorMessage name="UserName"></ErrorMessage></dd>  
                            <dt>Email</dt>  
                            <dd><Field name="Email" type="text"></Field></dd>  
                            <dd className="text-danger">  
                                <ErrorMessage name="Email"></ErrorMessage>  
                            </dd>  
                            <dt>Age</dt>  
                            <dd><Field name="Age" type="text"></Field></dd>  
                            <dd className="text-danger">  
                                <ErrorMessage name="Age"></ErrorMessage>  
                            </dd>  
                            <dt>City</dt>  
                            <dd><Field name="City" as="select">  
                                 <option>Delhi</option>  
                                 <option>Hyd</option>  
                                </Field></dd>  
                        </dl>  
                         <button disabled={(fields.isValid)?false:true} >Register</button>  
                       </div>  
                        }    
                    </Form>  
                     
                }  
            </Formik>  
        </div>  
    )  
}**