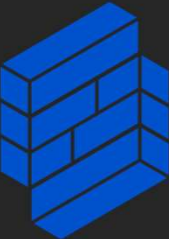

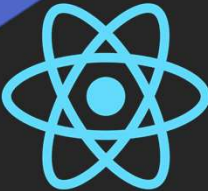


Venkatesh Mogili
#WebGuru


తెలుగు లో
PART-1




React FORMIK
Crash Course

Contents

- Introduction to Formik
- Formik Installation
- Form State, Validation, Submission
- Schema Validation with Yup
- Reducing Boilerplate
- Formik Core Components
- Formik Advanced Components
- Summary

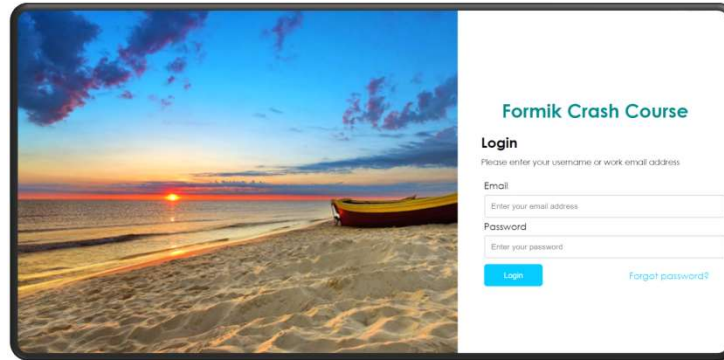


Venkatesh Mogili
#WebGuru



Final Output

- By the end of this course, you can see this Login form, Registration Form and Enrollment Forms with fully functional using Formik Library.



Venkatesh Mogili
#WebGuru



Introduction to Formik

- Formik is a small group of React components and hooks for building forms in React and React Native. It helps with the three most annoying parts:

- 1) Getting values in and out of form state
- 2) Validation and error messages
- 3) Handling form submission

- *As Formik is using context api, it doesn't need any state management libraries like redux or flux.*



Venkatesh Mogili
#WebGuru

"Build forms in React, without the tears."



Formik Installation

- Create a new react application:
 - `npx create-react-app formik-crash-course`
- Install formik:
 - `npm i formik`
 - OR
 - `yarn add formik`
- `import {useFormik} from 'formik';`
- Reference: <https://www.npmjs.com/package/formik>



Venkatesh Mogili
#WebGuru



FORMIK

Formik Feedback

Your Email

Your Name

Submit

Powered by  FORMIUM



Venkatesh Mogili
#WebGuru



FORMIK

Form State, Validation, Submission

- **useFormik** – a hook to use the formik library.
- **Form State:**
- **initialValues** – an object to define initial values of the form.
- **values** – values of the form for each input.
- **name** – a very important attribute to get and set the state values.
- **value** – state value
- **onChange** – input event for detecting input changes.



Venkatesh Mogili
#WebGuru



Form State, Validation, Submission

- **Form Validation:**
- **validate** – a method which takes values as an argument to handle validation for each input field.
- **errors** – errors of the form for each input.
- **onBlur** – input event for detecting input touch
- **Form Submission:**
- **onSubmit** – a method which takes values as an argument to handle form submission



Venkatesh Mogili
#WebGuru



Schema Validation with Yup

- Install yup:
- `npm i yup`
- OR
- `yarn add yup`
- `import * as Yup from 'yup';`
- Reference: <https://www.npmjs.com/package/yup>



Venkatesh Mogili
#WebGuru



FORMIK

Schema Validation with Yup

- **validationSchema** – property used for form level validation schema using Yup.
- **Few import Yup validation methods:**
- `Yup.object({key:validation})`
- `Yup.string().required("Validation Message");`
- `Yup.string().email("Invalid email").required("Required");`
- `Yup.number().required().positive().integer();`
- `Yup.string().url();`
- `Yup.date().required().nullable();`
- `Yup.number().min(18)`
- `Yup.array().required();`
- **Custom Validation:** `Yup.string().matches(RegularExpression,"Error Message");`



Venkatesh Mogili
#WebGuru



FORMIK

Reducing Boilerplate

- **getFieldProps()** - it returns to you the exact group of onChange, onBlur, value, checked for a given field. You can then spread that on an input, select, or textarea.

getFieldProps() = onChange, onBlur and value.



Venkatesh Mogili
#WebGuru



FORMIK

Formik Components

- **Formik has 4 main components.**
- **Formik** – It is a replacement for **useFormik()** hook.
- **Form** – it is a form to wrap the form elements. **Simply form tag.**
- **Field** – input fields with **name prop as mandatory**. We can give custom inputs by using “as” prop or “**component**” prop. We can also do field level validation.
- **ErrorMessage** – **name prop is mandatory** and **render** or **component prop can be used to display the custom error message** with style.



Venkatesh Mogili
#WebGuru



FORMIK

Formik Components Advanced

- **Render Props Pattern**
- **Nested Objects**
- **Arrays**
- **Advanced Formik Components:**
- **FieldArray**
- **FastField**
- Validation runs in **1.if any input changes, 2.when input touched, 3.on page load, 4.when form submitted**. We can restrict it with **validateOnChange** and **validateOnBlur** props.
- Field level validation using **validate** prop in Field component.



Venkatesh Mogili
#WebGuru



FORMIK

Formik Components Advanced

- **Manual Triggering Validation**
- ✓ **validateField(fieldName)**
- ✓ **validateForm()**
- ✓ **setFieldTouched(fieldName)**
- ✓ **setTouched({key:value})**
- **Disabling submit button:**
- ✓ Case-1: after clicking submit button if the form is invalid (!isValid)
- ✓ Case-2: On page load (validateOnMount)
- ✓ Case-3: On filling mandatory fields (!dirty & isValid)
- ✓ Case-4: On pre populating default data
- ✓ Case-5: while form submission is in progress
- **Load Saved Data and Reset Form**



Venkatesh Mogili
#WebGuru



FORMIK

Summary

- ✓ We have seen how to save the form state, validation and form submission in traditional way and with formik components.
- ✓ We have learned Yup validation library.
- ✓ We have learned how to use formik core and advanced components like Formik, Form, Field, ErrorMessage, FieldArray and FastField.
- ✓ We have learned how to trigger the validation manually with different validation methods and we have seen all the cases where disable option will be used in the forms.
- ✓ Finally, we have learned how to load the saved data and how to reset the form.



Venkatesh Mogili
#WebGuru



Part-2 Contents...

- Formik With Reusable Components
- Formik with Third Party Libraries
- Formik with Material UI
- Formik Practical Forms
- Summary



Venkatesh Mogili
#WebGuru



Don't forget to Share and Subscribe



Check out other courses



<https://vmtraining.netlify.app>



VWatch



IMplement



Try Experiments

VM Online Training



Thank you for watching



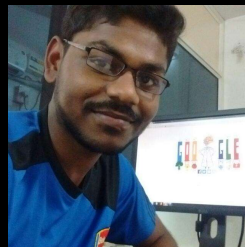
Subscribe

and



Stay Tuned

For more updates...



VENKATESH MOGILI

#Web Guru

