Exercise (Instructions): Controlled Form Validation

Objectives and Outcomes

In this exercise you will be introduced to simple form validation for controlled forms in React. At the end of this exercise you will be able to:

- Configure and perform simple form validation for your controlled forms

Simple Form Validation

- Open *ContactComponent.js* and update it as follows to introduce the support to track form errors and perform validation:

```
. . .
import { Breadcrumb, BreadcrumbItem, Button, Form, FormGroup, Label, Input, Col,
  Row, FormFeedback } from 'reactstrap';
. . .

class Contact extends Component {

    constructor(props) {
        super(props);

        this.state = {
            firstname: '',
            lastname: '',
            telnum: '',
            email: '',
            agree: false,
            contactType: 'Tel.',
            message: '',
            touched: {
                firstname: false,
                lastname: false,
                telnum: false,
                email: false
            }
        }
        this.handleSubmit = this.handleSubmit.bind(this);
        this.handleInputChange = this.handleInputChange.bind(this);
        this.handleBlur = this.handleBlur.bind(this);
    }

    . . .

    handleBlur = (field) => (evt) => {
        this.setState({
            touched: { ...this.state.touched, [field]: true }
        });
    }

    validate(firstname, lastname, telnum, email) {
        const errors = {
            firstname: '',
            lastname: '',
            telnum: '',
            email: ''
        };

        if (this.state.touched.firstname && firstname.length < 3)
            errors.firstname = 'First Name should be >= 3 characters';
        else if (this.state.touched.firstname && firstname.length > 10)
            errors.firstname = 'First Name should be <= 10 characters';

        if (this.state.touched.lastname && lastname.length < 3)
            errors.lastname = 'Last Name should be >= 3 characters';
        else if (this.state.touched.lastname && lastname.length > 10)
            errors.lastname = 'Last Name should be <= 10 characters';

        const reg = /^\d+$/;
        if (this.state.touched.telnum && !reg.test(telnum))
            errors.telnum = 'Tel. Number should contain only numbers';

        if(this.state.touched.email && email.split('').filter(x => x === '@'
          ).length !== 1)
            errors.email = 'Email should contain a @';

        return errors;
    }

    . . .

   render() {
        const errors = this.validate(this.state.firstname, this.state.lastname,
          this.state.telnum, this.state.email);

        . . .
```

- Now that we have introduced some functions that can be used for form validation, let us update the form itself to make use of these as follows:

```
. . .

                        <FormGroup row>
                            <Label htmlFor="firstname" md={2}>First Name
                              </Label>
                            <Col md={10}>
                                <Input type="text" id="firstname" name
                                  ="firstname"
                                    placeholder="First Name"
                                    value={this.state.firstname}
                                    valid={errors.firstname === ''}
                                    invalid={errors.firstname !== ''}
                                    onBlur={this.handleBlur('firstname')}
                                    onChange={this.handleInputChange} />
                                <FormFeedback>{errors.firstname}
                                  </FormFeedback>
                            </Col>
                        </FormGroup>
                        <FormGroup row>
                            <Label htmlFor="lastname" md={2}>Last Name
                              </Label>
                            <Col md={10}>
                                <Input type="text" id="lastname" name
                                  ="lastname"
```

```
                                placeholder="Last Name"
                                value={this.state.lastname}
                                valid={errors.lastname === ''}
                                invalid={errors.lastname !== ''}
                                onBlur={this.handleBlur('lastname')}
                                onChange={this.handleInputChange} />
                            <FormFeedback>{errors.lastname}
                              </FormFeedback>
                        </Col>
                    </FormGroup>
                    <FormGroup row>
                        <Label htmlFor="telnum" md={2}>Contact Tel
                          .</Label>
                        <Col md={10}>
                            <Input type="tel" id="telnum" name="telnum"
                                placeholder="Tel. Number"
                                value={this.state.telnum}
                                valid={errors.telnum === ''}
                                invalid={errors.telnum !== ''}
                                onBlur={this.handleBlur('telnum')}
                                onChange={this.handleInputChange} />
                            <FormFeedback>{errors.telnum}</FormFeedback>
                        </Col>
                    </FormGroup>
                    <FormGroup row>
                        <Label htmlFor="email" md={2}>Email</Label>
                        <Col md={10}>
                            <Input type="email" id="email" name="email"
                                placeholder="Email"
                                value={this.state.email}
                                valid={errors.email === ''}
                                invalid={errors.email !== ''}
                                onBlur={this.handleBlur('email')}
                                onChange={this.handleInputChange} />
                            <FormFeedback>{errors.email}</FormFeedback>
                        </Col>
                    </FormGroup>
                    . . .
```

- You can now test your form by typing in invalid input and check how the form validation works.
- Save all the changes and do a Git commit with the message "Controlled Form Validation"

Conclusions

In this exercise you have learnt about doing simple form validation for controlled forms in React.

Mark as completed