Phase 2 blog post:

Journey to the Lonely Mountain

We’ve left the Shire and have an important job: defeat the dragon, Smaug, so we can free the people of Lake-town and return the lost dwarven treasure, the team will need a burglar of exceptional skill. However, exceptional skill won’t be enough – we’ll need some tools to up our game. Luckily, we know a bit of HTML and Javascript to get us started on our quest, but we’ll need the ring of invisibility to help us get through the Mirkwood to the Lonely Mountain and Smaug’s lair!

React, a Javascript library, is a bit like the ring of invisibility. We already know how to write Javascript ourselves (be a burglar) but our powers are enhanced with additional tools. React is a declarative, component-based library that returns JSX (an XML-like syntax) such that complex webpages can be broken down into smaller components. With these components, different aspects of a webpage’s functionality can be organized and siloed, allowing for increased complexity while minimizing the costly re-rendering that can accommodate dynamic data.

React components are organized into parent-child-sibling relationships in the same manner as Javascript elements. However, React accommodates dynamic data by monitoring the value(s) of components’ state(s) using a virtual DOM. React ‘State’ is a built-in object designed to contain data or information about a component. When a state changes, the new state value can be compared to the existing value before finding the least expensive way to update the actual DOM using the new state, after which the component containing the state can be re-rendered. Because of this, React can update state more or less automatically in the affected component without forcing the browser to recalculate the position of everything on the webpage.

Monitoring state in (effectively) real-time has powerful consequences. With Javascript alone, a task like validating user inputs to a form (such as only allowing characters or digits to be entered) might have to wait until after the user has submitted the form. With React state, the state of the user-inputs can be updated anytime there’s a change to the inputs, regardless of whether or not the form has been submitted. This means the input can be validated before the form is submitted, and before any information is persisted to the backend unnecessarily. The process for creating a form component that checks the validity of form inputs before form submission might look something like this:

// import useState

>> import React, { useState } from "react";

// create React form component and set initial state

>> function myForm() {

const [formData, setFormData] = useState({

name: “”

phoneNumber: “”

email: “”

});

// function to handle changes to the form input(s) and update the form data state

>> function handleChange(event) {

setFormData({...formData, [event.target.name]: event.target.value});

}

// functions to validate form inputs

// Name

>> function checkName(formData.name) {

// check if form name is composed only of characters

}

// Phone Number

>> function checkPhoneNumber(formData.phoneNumber) {

// check if form phone number is composed only of digits

}

// return form JSX

>> return (

<section>

<h1>Form</h1>

<form>

<label>

Name:

<input

type="text"

name="name"

value={formData.name}

onChange={handleChange}

checkName={checkName}

/>

</label>

<label>

Phone Number:

<input

type="text"

name="phoneNumber"

value={formData.phoneNumber}

onChange={handleChange}

checkPhoneNumber={checkPhoneNumber}

/>

</label>

<label>

Email:

<input

type="text"

name="email"

value={formData.email}

onChange={handleChange}

/>

</label>

</form>

</section>

);

}

// export React component so other components can import it, if necessary

export default myForm;

Wow, that was much faster than it would have been for us to try to write Javascript to check the form inputs “live.” And, as a bonus, all of these checks are contained within our form component and are therefore quarantined from other components on the webpage. That way we can separate tasks and break features down into smaller, bite-sized chunks.

With the help of our new frontend tools, we’ve made our way through the Mirkwood to the Lonely Mountain. It’s time to learn how the backend works so we have all the skills we need to defeat Smaug and free the people of Lake-town!