Jomarie E. Pendon

BSIS 4B

PROFE8

Certificate 1

Verifiable linked: <https://www.freecodecamp.org/certification/fcc441fc66c-4cb0-4d57-bf6f-b272ee14d3e2/front-end-development-libraries>

Front End Development Libraries

Course Content:

Bootstrap

● Use Responsive Design with Bootstrap Fluid Containers

● Make Images Mobile Responsive

● Center Text with Bootstrap

● Create a Bootstrap Button

● Create a Block Element Bootstrap Button

● Taste the Bootstrap Button Color Rainbow

● Call out Optional Actions with btn-info

● Warn Your Users of a Dangerous Action with btn-danger

● Use the Bootstrap Grid to Put Elements Side By Side

● Ditch Custom CSS for Bootstrap

● Use a span to Target Inline Elements

● Create a Custom Heading

● Add Font Awesome Icons to our Buttons

● Add Font Awesome Icons to all of our Buttons

● Responsively Style Radio Buttons

● Responsively Style Checkboxes

● Style Text Inputs as Form Controls

● Line up Form Elements Responsively with Bootstrap

● Create a Bootstrap Headline

● House our page within a Bootstrap container-fluid div

● Create a Bootstrap Row

● Split Your Bootstrap Row

● Create Bootstrap Wells

● Add Elements within Your Bootstrap Wells

● Apply the Default Bootstrap Button Style

● Create a Class to Target with jQuery Selectors

● Add id Attributes to Bootstrap Elements

● Label Bootstrap Wells

● Give Each Element a Unique id

● Label Bootstrap Buttons

● Use Comments to Clarify Code

jQuery

● Learn How Script Tags and Document Ready Work

● Target HTML Elements with Selectors Using jQuery

● Target Elements by Class Using jQuery

● Target Elements by id Using jQuery

● Delete Your jQuery Functions

● Target the Same Element with Multiple jQuery Selectors

● Remove Classes from an Element with jQuery

● Change the CSS of an Element Using jQuery

● Disable an Element Using jQuery

● Change Text Inside an Element Using jQuery

● Remove an Element Using jQuery

● Use appendTo to Move Elements with jQuery

● Clone an Element Using jQuery

● Target the Parent of an Element Using jQuery

● Target the Children of an Element Using jQuery

● Target a Specific Child of an Element Using jQuery

● Target Even Elements Using jQuery

● Use jQuery to Modify the Entire Page

SASS

● Store Data with Sass Variables

● Nest CSS with Sass

● Create Reusable CSS with Mixins

● Use @if and @else to Add Logic To Your Styles

● Use @for to Create a Sass Loop

● Use @each to Map Over Items in a List

● Apply a Style Until a Condition is Met with @while

● Split Your Styles into Smaller Chunks with Partials

● Extend One Set of CSS Styles to Another Element

React

● Create a Simple JSX Element

● Create a Complex JSX Element

● Add Comments in JSX

● Render HTML Elements to the DOM

● Define an HTML Class in JSX

● Learn About Self-Closing JSX Tags

● Create a Stateless Functional Component

● Create a React Component

● Create a Component with Composition

● Use React to Render Nested Components

● Compose React Components

● Render a Class Component to the DOM

● Write a React Component from Scratch

● Pass Props to a Stateless Functional Component

● Pass an Array as Props

● Use Default Props

● Override Default Props

● Use PropTypes to Define the Props You Expect

● Access Props Using this.props

● Review Using Props with Stateless Functional Components

● Create a Stateful Component

● Render State in the User Interface

● Render State in the User Interface Another Way

● Set State with this.setState

● Bind 'this' to a Class Method

● Use State to Toggle an Element

● Write a Simple Counter

● Create a Controlled Input

● Create a Controlled Form

● Pass State as Props to Child Components

● Pass a Callback as Props

● Use the Lifecycle Method componentWillMount

● Use the Lifecycle Method componentDidMount

● Add Event Listeners

● Optimize Re-Renders with shouldComponentUpdate

● Introducing Inline Styles

● Add Inline Styles in React

● Use Advanced JavaScript in React Render Method

● Render with an If-Else Condition

● Use && for a More Concise Conditional

● Use a Ternary Expression for Conditional Rendering

● Render Conditionally from Props

● Change Inline CSS Conditionally Based on Component State

● Use Array.map() to Dynamically Render Elements

● Give Sibling Elements a Unique Key Attribute

● Use Array.filter() to Dynamically Filter an Array

● Render React on the Server with renderToString

Redux

● Create a Redux Store

● Get State from the Redux Store

● Define a Redux Action

● Define an Action Creator

● Dispatch an Action Event

● Handle an Action in the Store

● Use a Switch Statement to Handle Multiple Actions

● Use const for Action Types

● Register a Store Listener

● Combine Multiple Reducers

● Send Action Data to the Store

● Use Middleware to Handle Asynchronous Actions

● Write a Counter with Redux

● Never Mutate State

● Use the Spread Operator on Arrays

● Remove an Item from an Array

● Copy an Object with Object.assign

React and Reux

● Getting Started with React Redux

● Manage State Locally First

● Extract State Logic to Redux

● Use Provider to Connect Redux to React

● Map State to Props

● Map Dispatch to Props

● Connect Redux to React

● Connect Redux to the Messages App

● Extract Local State into Redux

● Moving Forward from Here

