**UI Component Libraries**

When you start designing User Interfaces (UI) using JS frameworks, you start realizing that everything revolves around UI Components. You need components for Navigation Bar, Tables, Cards, Buttons, Forms, Modals, etc. We have seen in our React examples that designing good UI components takes a lot of time and requires some artistic ability, which I personally do not have ☺ Observing the need for nicely designed, reusable UI components, people have come up with UI Component Libraries. In this lecture, we will look into some of the most widely used UI Component libraries in the market, some of which are free and some are commercial. Here is a non-comprehensive list:

1. React-Bootstrap (<https://react-bootstrap.github.io/>): Free React Components. There are also Bootstrap components Angular (<https://ng-bootstrap.github.io/>), Vue (<https://bootstrap-vue.org/>), and Svelte (<https://sveltestrap.js.org/>)
2. Material UI (<https://mui.com/>): Free React Components. There are also Material Components for Angular (<https://material.angular.io/>), Vue (<https://next.vuetifyjs.com/en/>), and Svelte (<https://sveltematerialui.com/>).
3. Ant Design (<https://ant.design/>): Free React Components. There are also Ant components for Angular (<https://ng.ant.design/>), Vue (<https://antdv.com/>). Ant components for Svelte is currently under development (<https://www.npmjs.com/package/@tommywalkie/ant-design-svelte>).
4. Prime Faces (<https://www.primefaces.org/>): Commercial components for Angular, React, Vue, Java Server Faces (JSF)
5. DevExtreme (<https://js.devexpress.com/>): Commercial components for Angular, React, Vue, jQuery Components
6. Chakra UI (<https://chakra-ui.com/>): Free? Components for React. There are also Chakra UI components for Vue (<https://vue.chakra-ui.com/>), and Svelte (<https://chakra-svelte.vercel.app/>).

For a more comprehensive list of UI libraries for Angular, React and Vue, look at the following page: <https://dev.to/haycuoilennao19/34-ui-libraries-for-react-vue-and-angular-525l>

**React-Bootstrap [**<https://react-bootstrap.github.io/>**]**

React-Bootstrap is a free React component library built on top of the Bootstrap CSS. If you are comfortable with Bootstrap CSS, it should be quite easy for you to get started with React-Bootstrap.

**Installation**

After creating a boilerplate React project, go to the project directory and install React-Bootstrap as described in <https://react-bootstrap.github.io/getting-started/introduction/>:

|  |
| --- |
| % npm install react-bootstrap bootstrap |

If you want to try out React-Bootstrap without creating a stand-alone app, you can simply include the React-bootstrap javascript library into your HTML file and directly use it as follows:

|  |
| --- |
| <!-- Load React Bootstrap Library -->  <script  src="https://cdn.jsdelivr.net/npm/react-bootstrap@next/dist/react-bootstrap.min.js" crossorigin>  </script> |

You also need to include the React and ReactDOM JS libraries and the babel compiler for this to work:

|  |
| --- |
| <script src="https://unpkg.com/react@latest/umd/react.development.js" crossorigin="anonymous"></script>  <script src="https://unpkg.com/react-dom@latest/umd/react-dom.development.js"></script>  <script src="https://unpkg.com/babel-standalone@latest/babel.min.js" crossorigin="anonymous"></script> |

Finally, you need to include the Bootstrap CSS library as follows:

|  |
| --- |
| <link  rel="stylesheet"  href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"  integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"  crossorigin="anonymous"  /> |

Here is your first React-Bootstrap application: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index1.html

**Components**

React-Bootstrap consists of many ready-to-use React components. You can find all components by selecting the “Components” menu item on the left side. Here is the page for the Button component: https://react-bootstrap.github.io/components/buttons/. By scrolling down the page, you can find different sorts of Button components and how to use them.

Here is an example showing how to use Buttons: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index2.html. In this example we also made use of the Stack component (<https://react-bootstrap.github.io/layout/stack/>) to vertically stack different components and also leave some gap between them (gap={3}). It is also possible to horizontally stack different components by specifying direction="horizontal" (<Stack direction="horizontal" gap={3}>).

Here is another example that makes use of the Card component: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index3.html

**Grid System**

As you already know, Bootstrap has a 12-column grid system for responsive designs. In order to use the grid system, you need to make use of the **Row** and **Col** components: <https://react-bootstrap.github.io/layout/grid/>. In the following example, we converted the previous Card example into a responsive design, where each card occupies all 12 columns for extra small screens, 6 columns for medium screens and 4 columns for large screens and up:

|  |
| --- |
| <Container>  <Row>  {notes.map(note => (  <Col xs={12} md={6} lg={4} key={note.id}>  <NoteCard note={note}/>  </Col>  ))}  </Row>  </Container> |

Look at 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index4.html

In fact, we do not need to specify xs={12} because Bootstrap is a mobile-first design. That is, when you do not specify how much space a component should occupy on the grid row, it is assumed to be 12 columns, i.e., the full width of the row. You then need to specify how much space a component should occupy as the screen gets larger. In this example, we want each card to occupy 6 columns for medium screen sizes (768-992px). When we reach the large screen size (>=992px), we want each Card component to occupy 4 columns so that we see 3 Cards on the same row. So, even if we specified <Col md={6} lg={4}>, we would have gotten the same result.

**Forms**

Forms are an essential part of any non-trivial App. Here is the link to the React-Bootstrap form page (<https://react-bootstrap.github.io/forms/overview/>). The <Form> component wraps all form elements. The <Form.Group> component wraps a form control with proper spacing, along with support for a label, help text, and validation state. <Form.Label> is used to set a label for the form element and <Form.Control> component renders a form control element with Bootstrap styling. In order to access the value of an input field (Form.Control), set controlId on <Form.Group>. Here is an example with <Form>, <Form.Group>, <Form.Label> and <Form.Control> components:

Look at 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index5.html

In order to access the values of the input fields after the form is successfully submitted, you can install a callback function (onSubmit) on the <Form> component and retrieve the values of the input fields. The following example shows how this can be done: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index6.html

Here is another example with more form controls: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index7.html

**Form Layouts:** So far all our <Form.Group> components occupied 100% of the <Form> container. It is possible to change this layout by using the “grid” layout classes <**Row**> and <**Col**>. In the following example, we have two <Form.Group> elements. Each element occupies only 50% of the row for medium and large screen sizes (>=768px), but 100% of the row for small screen sizes (<768px): 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index8.html

**Horizontal Form Elements:** You can create horizontal forms with the grid by adding as={Row} to form groups and using Col to specify the width of your labels and controls. Be sure to add the column prop to your FormLabels as well so they’re vertically centered with their associated form controls. 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index9.html

**Form Validation** (<https://react-bootstrap.github.io/forms/validation/>): Form validation follows the same logic as the form validation with Bootstrap CSS. Here is an example: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index10.html

**Modals** (<https://react-bootstrap.github.io/components/modal/>)

Modals are a very important part of any non-trivial Web app. Here is an example that shows how you can create a Modal, open, and close it: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index11.html

**Pagination** (<https://react-bootstrap.github.io/components/pagination/>)

React-Bootstrap also provides support for pagination display. Here is an example that has 5 pages and also handles page changes by listening to the button click events: 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index12.html

Here is a more complex example that shows how you can handle pagination when you have too many pages (20 in this example): 10-UI-Comp-Libs/ReactBootstrap/01-FirstEx/index13.html

**Stand-alone React-Bootstrap App using React-Bootstrap**

So far all our examples used the online React & Babel compilers to work. Here is an example that installs React-Bootstrap and uses it in a stand-alone React app. This app shows examples of several React-Bootstrap components: 10-UI-Comp-Libs/ReactBootstrap/02-CLI-Ex

**Material UI [**https://mui.com/**]**

Material UI is an open-source React component library that implements Google's Material Design. It includes a comprehensive collection of prebuilt components that are ready for use in production right out of the box. Material UI is completely free, and features a suite of customization options that make it easy to implement your own custom design system on top of our components.

**Installation**

After creating your boilerplate React app, go to the app directory and install MUI as described in <https://mui.com/material-ui/getting-started/installation/>:

|  |
| --- |
| % npm install @mui/material @emotion/react @emotion/styled |

If you want to try out React-Bootstrap without creating a stand-alone app, you can simply include the Material UI javascript library into your HTML file and directly use it as follows:

|  |
| --- |
| <script  src="<https://unpkg.com/@mui/material@latest/umd/material-ui.development.js>" crossorigin="anonymous"> </script> |

You also need to include the React and ReactDOM JS libraries and the babel compiler for this to work:

|  |
| --- |
| <script src="https://unpkg.com/react@latest/umd/react.development.js" crossorigin="anonymous"></script>  <script src="https://unpkg.com/react-dom@latest/umd/react-dom.development.js"></script>  <script src="https://unpkg.com/babel-standalone@latest/babel.min.js" crossorigin="anonymous"></script> |

Finally, you need to include the CSS libraries for font support and icons if you use them as follows:

|  |
| --- |
| <!-- Fonts to support Material Design -->  <link  rel="stylesheet"  href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap"  />  <!-- Icons to support Material Design -->  <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons" /> |

Here is your first MUI application: 10-UI-Comp-Libs/MUI/01-FirstEx/index1.html

**Components**

MUI consists of many ready-to-use React components. You can find all components by selecting the “Components” menu item on the left side. Here is the page for the Button component: <https://mui.com/material-ui/react-button/>. By scrolling down the page, you can find different sorts of Button components and how to use them. For example, here is the code for “Contained” buttons:

|  |
| --- |
| import \* as React from 'react';  import Button from '@mui/material/Button';  import Stack from '@mui/material/Stack';  export default function ContainedButtons() {  return (  <Stack direction="row" spacing={2}>  <Button variant="contained">Contained</Button>  <Button variant="contained" disabled>  Disabled  </Button>  <Button variant="contained" href="#contained-buttons">  Link  </Button>  </Stack>  );  } |

Look at 10-UI-Comp-Libs/ MUI/01-FirstEx/index2.html

To find all “props” that you can use with a component, you can go to the Component API by selecting the “Component API” menu item on the left. Here is the Component API page for the Button component: <https://mui.com/material-ui/api/button/>

Here is another example that makes use of the Card component: 10-UI-Comp-Libs/MUI/01-FirstEx/index3.html

**Grid System**

As you already know, Material has a 12-column grid system for responsive designs. In order to use the grid system, you need to make use of the Grid component: <https://mui.com/material-ui/react-grid/>.

In the following example, we converted the previous Card example into a responsive design, where each card occupies all 12 columns for extra small screens, 6 columns up to medium screen size and 4 columns for large screens and up:

|  |
| --- |
| <Grid container spacing={3}>  {notes.map(note => (  <Grid item xs={12} md={6} lg={4}>  <NoteCard note={note} key={note.id}/>  </Grid>  ))}  </Grid> |

Look at 10-UI-Comp-Libs/MUI/01-FirstEx/index4.html

In fact, we do not need to specify xs={12} because the Material is a mobile-first design. That is, when you do not specify how much space a component should occupy on the grid row, it is assumed to be 12 columns, i.e., the full width of the row. You then need to specify how much space a component should occupy as the screen gets larger. In this example, we want each card to occupy 6 columns for medium screen sizes and up (900px and up). When we reach the large screen size (1200px), we want each Card component to occupy 4 columns so that we see 3 Cards on the same row. So, even if we specified <Grid item md={6} lg={4}>, we would have gotten the same result.

**Theming**

The theme specifies the color of the components, darkness of the surfaces, level of shadow, appropriate opacity of ink elements, etc. By default, MUI comes with default themes (e.g., default values for all values potential themes). For example, if you go to <https://mui.com/material-ui/customization/palette/> you can see the default values for all colors in the palette. Themes let you apply a consistent tone to your app.

Sometimes you want to change the default values in the themes in order to meet the specific needs of your business or brand. In order to do that you first create a theme object using “createTheme” hook and then pass this new theme to all components using the “ThemeProvider” component. ThemeProvider relies on the context feature of React to pass the theme down to the components, so you need to make sure that ThemeProvider is a parent of the components you are trying to customize. The following example illustrates how we can change the primary and secondary colors in the palette and make our Components use these new colors.

Look at 10-UI-Comp-Libs/MUI/01-FirstEx/index5.html

**Net Ninja’s Notes App in React using Material UI and Router v5 and JSON-Server**

The Net Ninja’s tutorial on MUI: (<https://www.youtube.com/playlist?list=PL4cUxeGkcC9gjxLvV4VEkZ6H6H4yWuS58>) (<https://github.com/iamshaunjp/material-ui-tut>). Here, a Card-based note system is built.

In order to run these projects, you need to install JSON-server (<https://www.npmjs.com/package/json-server>) and run it with the notes database stored in data/db.json as follows:

|  |
| --- |
| % json-server --watch data/db.json –p 8000 |

Notice that the JSON server runs at port 8000 (-p 8000). By default it runs at 3000, which collides with the HTTP server that React starts, so we change the port number to 8000.

After starting the JSON server, we can now start the React app by typing:

|  |
| --- |
| % npm start |

Look at 10-UI-Comp-Libs/MUI/02-NetNinja-NoteCard1

Look at 10-UI-Comp-Libs/MUI/03-NetNinja-NoteCard2

**Ant Design [**https://ant.design/**]**

Ant Design is a React UI library named ‘antd’ that contains a set of high quality components and demos for building rich, interactive user interfaces. Here are some of the important features of AntD:

* Enterprise-class UI designed for web applications.
* A set of high-quality React components out of the box.
* Written in TypeScript with predictable static types.
* Whole package of design resources and development tools.
* Internationalization support for dozens of languages.
* Powerful theme customization in every detail.

**Installation**

After creating your boilerplate react project using “npx create-react-app project”, go to the project directory and install MUI as described in https://ant.design/docs/react/introduce:

|  |
| --- |
| % npm install antd |

If you want to try out React-Bootstrap without creating a stand-alone app, you can simply include the AntD javascript library into your HTML file and directly use it as follows:

|  |
| --- |
| <script  src="https://cdnjs.cloudflare.com/ajax/libs/antd/4.23.2/antd.min.js" integrity="sha512-ktMcWbnTz8JSq52/Y8fdaSL28qmPKIJMXQqgwkEJGge1oWZ599hssJFlA++F/PmPMPkp4aWZIgvDLp9Wt2PQZw==" crossorigin="anonymous" referrerpolicy="no-referrer">  </script> |

Here is your first AntD application: 10-UI-Comp-Libs/AntD/01-FirstEx/index1.html

Some AntD buttons: 10-UI-Comp-Libs/AntD/01-FirstEx/index2.html

Some antD Cards: 10-UI-Comp-Libs/AntD/01-FirstEx/index3.html

A React App created with react CLI using a antD Table Component: 10-UI-Comp-Libs/AntD/02-CLI-Ex

**Prime Faces (**<https://www.primefaces.org/>**)**

Prime Faces is a commercial UI Component Library with some free Components. The advantage of Prime Faces is that every Component is available for all major JS libraries. That is, every Component is available for Angular, React, Vue and Java Server Faces (JSF), which could be very beneficial if you have a project that uses a combination of these technologies. By using the same Component, you can get an application with the same look and feel across different platforms.

**Installation (**<https://www.primefaces.org/primereact/setup/>**)**

As with the other UI Component Libraries, you can use Prime Faces by installing it with npm (recommended method) or by directly including it in your HTML. Here is how you install it using npm:

|  |
| --- |
| npm install primereact primeicons primeflex |

Then start using the Components from your React apps as follows:

|  |
| --- |
| // import { ComponentName } from 'primereact/{componentname}';  import { Dialog } from 'primereact/dialog';  import { Button } from 'primereact/button'; |

Also, you may need to include the CSS dependencies:

|  |
| --- |
| import "primereact/resources/themes/lara-light-indigo/theme.css"; //theme  import "primereact/resources/primereact.min.css"; //core css  import "primeicons/primeicons.css"; //icons |

Alternatively, you can put the component library JS directly in your HTML file and use it (not recommended).

Here is your first Prime Faces application showing Buttons: 10-UI-Comp-Libs/PrimeFaces/01-FirstEx/index1.html

Slider: 10-UI-Comp-Libs/PrimeFaces/01-FirstEx/index2.html

Data Table (<https://www.primefaces.org/primereact/datatable/>) with pagination example with React CLI.

10-UI-Comp-Libs/PrimeFaces/02-CLI-Ex

**DevExtreme [**https://js.devexpress.com**]**

DevExtreme is another commercial UI Component library. Similar to Prime Faces, it has Components for most major JS libraries including Angular, React, Vue, and also for jQuery.

**Installation (**<https://js.devexpress.com/Documentation/Guide/React_Components/Add_DevExtreme_to_a_React_Application/>**)**

Create a boiler-plate React app, then install devextreme library as follows:

|  |
| --- |
| % npx -p devextreme-cli devextreme add devextreme-react |

Here is an example that makes use of Chart, PieChart, DataGrid, and Button components from DevExtreme:

10-UI-Comp-Libs/DevExtreme/01-CLI-Ex