REACT

Introduction to client side frameworks

- The client-side framework is usually a JavaScript library and runs in a Web browser, such as React, Angular and Vue.
- These libraries are frameworks in that they build higher-level APIs for client-side programming on top of the standard and proprietary APIs that web browsers offer.
- A framework is a library that offers opinions about how software gets built.
- using a framework will reduce the complexity while coding and managing stuff.
- Most major frameworks encourage developers to abstract the different parts of their user interfaces into components which are maintainable, reusable chunks of code that can communicate with one another.
- When routing is handled by a client application it is called client-side routing.

Components

- A Component is one of the core building blocks of React.
- Components are nothing but the reusable part of the (UI) user interface.
- In React, there are two types of components: Functional components and class components.
- Functional components are simply javascript functions. We can create a functional component in React by writing a javascript function.
- The class components are a little more complex than the functional components.
- Properties, or props, are external data that a component needs to render.
- A state is a JavaScript object that stores a component's dynamic data and determines the component's behaviour.
- Because the state is dynamic, it enables a component to keep track of changing information in between renders and for it to be dynamic and interactive.
- useState() is a React hook which, given initial data value, will keep track of that value as it is updated.

Getting Started with Reactjs

- React is a JavaScript library that forces you to think in terms of

components.
- In 2011, Facebook engineer Jordan Walke created React JS specifically to improve UI development.
- React comes with two key features that add to its appeal for JavaScript developers: JSX and Virtual DOM.
- JSX stands for JavaScript XML.
- JSX allows us to write HTML in React.
- JSX makes it easier to write and add HTML to React.
- React JS creates something called a Virtual DOM.
- The Virtual DOM (like the name implies) is a copy of the site's DOM, and React JS uses this copy to see what parts of the actual DOM need to change when an event happens (like a user clicking a button).

Installation and Setting Up

- Requirements for the reactjs applications are Nodejs , Any ide, Web browser, and create-react-app package.
- Nodejs is a javascript runtime environment built on top of chrome's v8 engine.
- The Node.js run-time environment includes everything you need to execute a program written in JavaScript.
- Create-react-app package which we will install from the node package manager.
- This package will do all the bootstrapping of a new reactjs project. It will create a boilerplate where we can work on.
- Install nodejs from the node js website.
- Open command prompt in windows or terminal in linux and type in `node -v`.
- Install create-react-app package from the node package manager `npm install -g create-react-app`
- npm is the package manager for node, -g means installing the package globally which will make the package available everywhere in your machine and the create-react-app is the package name.
- Create a react project folder with `npx create-react-app project_name`.
- To get inside the project folder type the command `cd project_name`.
- And to start the server type in the following in the terminal: `npm start`.

File structure

- The node_modules folder contains the files of packages that are

required to run the app (dependencies). This folder is managed by npm.
- Public folder is the entry folder of the project. All assets that can be directly accessed via urls are placed in this folder.
- The src directory is where we'll spend most of our time, as it is where the source code for our application lives.
- The App.css file contains the styles specific for the app component.
- The App.js file contains the code of the app component. This component is the root component of the app. That is the entire application is contained within this component.
- The Index.js file contains the code of the app component. This component is the root component of the app. That is the entire application is contained within this component.
- The App.test.ts file contains the test scripts of the app component.
- The .gitignore file contains the file paths that should not be included in the git repository.
- The package.json file contains information about our project that Node.js/npm uses to keep it organised.
- The package-lock.json file contains the dependencies of the react appThat are already installed.

JSX

- JSX stands for Javascript XML.
- It makes it easier for us to create templates.
- JSX allows us to use attributes with the HTML elements just like we do with normal HTML.
- But instead of using normal HTML naming conventions, JSX uses camelcase conventions for attributes.
- JSX allows us to write HTML in React.

State

- The state is an updatable structure that is used to contain data or information about the component and can change over time.
- The change in state can happen as a response to user action or system events.
- It is the heart of the react component which determines the behaviour of the component and how it will render.
- A state can be created with the help of `useState()` hook in a functional component.

Props

- Props are read-only components.

- It is an object which stores the value of attributes of a tag and works similar to the HTML attributes.
- It allows passing data from one component to other components.
- It is similar to function arguments and can be passed to the component the same way as arguments passed in a function.
- Props are immutable so we cannot modify the props from inside the component.

Form Handling

- Handling forms is about how we handle the data when it changes value or gets submitted.
- In HTML, form data is usually handled by the DOM.In React, form data is usually handled by the components.
- When the data is handled by the components, all the data is stored in the component state.
- We can control changes by adding event handlers in the onChange attribute.
- We can use the useState Hook to keep track of each input value and provide a "single source of truth" for the entire application.
- We can control the submit action by adding an event handler in the onSubmit attribute for the `<form>` tag.

Conditional Rendering

- In React, we can conditionally render the components in many ways.
- We can use the javascript operator to decide which component to render.
- Using the logical && operator we can conditionally render the react components.
- Another way to conditionally render the react component is by using the ternary operator.

Events

- React can perform actions based on the user events like onchange, onclick, mousehover , etc.,
- React events are written in camelCase format like onClick instead of onclick.
- React event handlers are written inside curly braces onClick = {eventhandler} instead of onClick = “ function()”.
- To pass an argument to the event handler use the arrow function.

Routing

- To add a router in the react application install the react router dom using npm.
- Run the following command to install the router `npm install react-router-dom`.
- Wrap the content inside the `<BrowserRouter>`. Then define our `<Routes>`.
- `<Route>` s can be nested inside the `<Routes>`.
- The `<Outlet>` renders the current selected route.
- `<Link>` is used to set the URL and keep track of browsing history.

Redux

- We learned about the state earlier. But the problem is that the state is only available inside that particular component.
- What if we need a state that is available across multiple components.
- The advantage of this is that we can access the same variable from any component in the app.
- To use redux in our react app we need to install the @reduxjs/toolkit package and react-redux package.
- We can install them with npm using the following command `npm install @reduxjs/toolkit react-redux`
- Next we will create a store to act as the state of the application.
- We use the configureStore hook provided in @reduxjs/toolkit to create a store.
- `reducer:{}` will contain all the slices of this store.
- We will use `Provider` to make the store available across the application.
- The store is set using the store property of the Provider. The store will be available to all components inside the Provider.
- So to make the store available across the application we will add the Provider around RouterProvider.
- Next step is to create slices for the store. A slice will be used to define an immutable variable and the functions(actions) used to modify this variable.
- `createSlice` is used to create a slice.
- name contains a string as a name for the slice.
- initialState contains the initial value of the slice variable.
- reducers contain the actions for the slice.
- The `action.payload` will contain the parameter passed when the action is dispatched.
- We will use the `useSelector`hook to get data from the store.
- Create a dispatch function using the `useDispatch` hook

- Call the actions as parameters of the dispatch function.

API handling with Reactjs and axios

- A RESTful API -- also referred to as a RESTful web service or REST API -- is based on representational state transfer (REST), an architectural style and approach to communications often used in web services development.
- An Http request is the method used to communicate with your server or an external API. It has 4 parts
- **Url/endpoint**
  This is the address of the API. This tells us where the request should be sent to. This is a must have for all Http requests.
- **Method**
  The method of the request indicates what it will be doing on the server. There are many Http Methods available, but most commonly used ones are GET, POST, PUT, PATCH, DELETE.
- **Body**
  Body contains the data to be passed using the request.
- **Headers**
  Headers contain the metadata of the request.
- Axios is a Javascript library used to make HTTP requests from node.js or XMLHttpRequests from the browser that also supports the ES6 Promise API.
- Axios is promise-based and thus we can take advantage of async and await for more readable asynchronous code.
- We can also intercept and cancel requests, and there's built-in client-side protection against cross-site request forgery.
- To install Axios run the following command in the terminal or cmd npm install axios - - save.
- Axios uses functions in the name of http method to send the request: axios.get();
- The url is passed as the first parameter.
- Body is passed as the second parameter. Note that body is not present for get and delete requests.
- Headers are passed inside the options object as the third parameter for the post, put, patch requests and second parameter for get and delete requests.
- These functions return a promise, so we will use the then method to handle the success response and the catch method to handle the error response.

Using API to create CRUD

- CRUD stands for Create, Read, Update, Delete.
- The CRUD operations can be performed with the help of axios and api.
- The useEffect() hook is used while fetching data from the api.

Authentication

- Authentication is the process of identifying the user accessing the site and restricting the content to certain users.
- We can't simply give access to a user. They need to provide a username and password.
- We will collect the username and password using a form.
- These are then sent to the login api.
- The api verifies whether the username and password is correct and if they are correct it returns a token.
- This token is used by the api to identify the user.
- Therefore this token needs to be sent with every request to the server that is available only for logged in users.
- Different apis have different ways in which they accept a token like header, query parameter etc.

Building and Deploying Project

- There are a few steps we have to follow for deployment. Let's go through the steps one by one.
- Run the following command to create an optimized build of the code `npm run build`.
- Now the next step is to deploy the generated files to a static host. There are a ton of static host service providers available like netlify, Amazon s3, Firebase hosting etc. For this project, I will be demonstrating the hosting with firebase.
- Now the final step is to deploy the web app to firebase. To do so follow the steps one by one:
- Inorder to deploy we first need to install the firebase cli. Run the following command in the terminal: `npm install -g firebase-tools`.
- login to the firebase cli tool with the gmail account to do that run the following command : `firebase login`
- Initialize firebase in our project to do that run the following command in the command line: `firebase init`
- Once the initialization is complete, enter the following command in the terminal to deploy the project to firebase hosting: `firebase deploy`.