Intro and Syllabus

Pre-Requisites

The prerequisites for this course are:

1. Basic knowledge of HTML and CSS.
2. JavaScript Concepts like event listeners, ES-6 concepts, etc.

Syllabus

The topics which are going to be covered in this course are:

1. **Introduction to React**
    - a. What is React?
    - b. What are components?
    - c. SPA v/s MPA
    - d. DOM and VDOM
    - e. Advantages of React
    - f. Basic Terminal Commands
2. **Javascript Concepts Revision** (Practice Module)
3. **How to create a React Project?**
    - a. create-react-app
    - b. Webpack, Babel
    - c. Import Exports
    - d. Folder Structure
    - e. React Developer Tools
    - f. JSX (JavaScript Extendable)
    - g. Conditional Rendering
4. **Mini Project: Shopping Cart**
    - a. Adding Styles
    - b. Creating a state
    - c. Event Handling
    - d. Update and Manage State using setState

e. React Props

f. Rendering a List, Mapping

g. Connecting to Firebase

h. How does Realtime DB work?

i. Component Lifecycle

   i. Mounting

   ii. Updating

   iii. Unmounting

j. Function chaining

k. Firebase

   i. Reading data

   ii. Adding data

   iii. Updating data

   iv. Deleting data

5. React Hooks

a. What are Hooks?

b. Why Hooks?

c. useState hook

d. useEffect hook

e. Custom Hooks

f. Rules of Hooks

6. Mini Project: Blog web app using hooks

a. App structure and setup

b. React Router

c. Pages - Home, Post Details, etc.

d. Deployment

e. Types of CSS Styling

   i. Inline

   ii. Internal

   iii. External

f. CSS modules

g. Styled components module

h. Dynamic Styling

i. Radium

7. **Major Project: Codeial**
    a. Setting up api
    b. Fetching data
    c. React Router
        i. Switch
        ii. Link
    d. Auth in SPA, Token
    e. Fragments
    f. Types of Routes
        i. Private
        ii. Public
        iii. Restricted
    g. Context
    h. Search Logic
8. **Redux**
    a. What is redux?
    b. When and Why use redux?
    c. Principles of Redux
    d. Redux Flow
    e. Pure Functions
    f. Reducers, Store, Actions
    g. Currying
    h. Middleware
    i. Async Actions: Thunk
    j. Context API
    k. Higher-Order Components
    l. Connecting React Components to redux

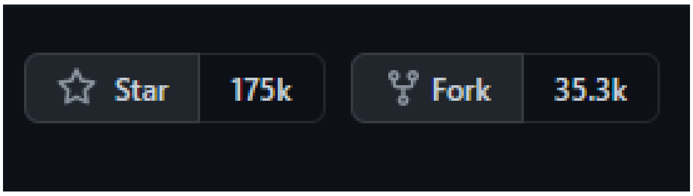
Why should you learn React?

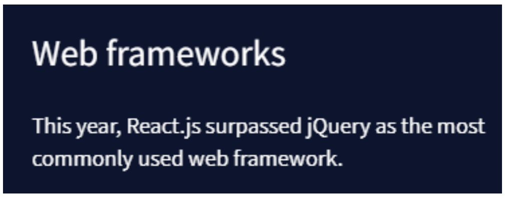
- **React** was originally created by **Facebook**, who now use it in many of their apps. Facebook continues to care for and improve React.
- React has a **massive community of dedicated developers**. The community helps to maintain and grow the **open-source library,** with everyone able to contribute their own components to the library.

  Github Repository: https://github.com/facebook/react
- Compared to other libraries, frameworks or programming languages, React is relatively **easy to learn**. It isn't a heavyweight framework like Angular, and it's intuitive.

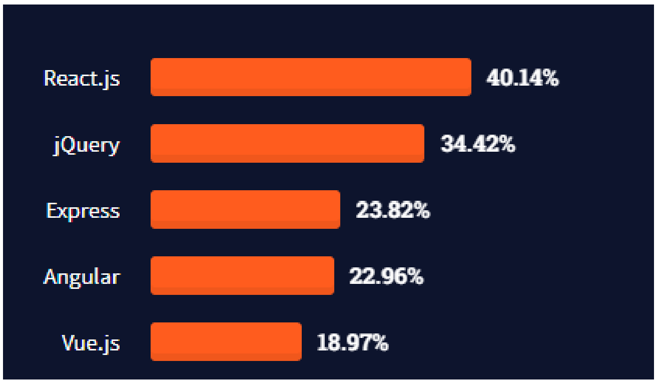
  Official Website: https://reactjs.org/
- React **saves you time and money** on development because it's component-based. You can break down an interface into reusable components that allow you to build dynamic user interfaces.
- Jobs you will get as a React developer in any of those companies will pay you a **high salary**.

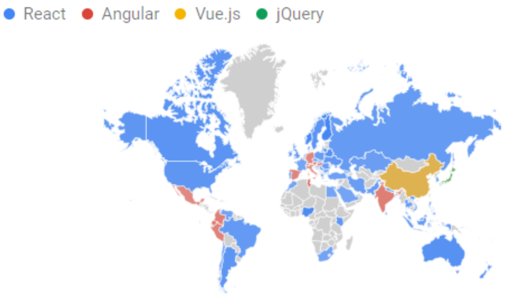
  In the US, the average salary for a React developer is 120K a year.




Web frameworks

This year, React.js surpassed jQuery as the most commonly used web framework.





Colour intensity represents percentage of searches LEARN MORE