ReactJS Program

ReactJS Curriculum

1. Why, what & how?
2. The problem we’re trying to solve
3. Libraries vs Frameworks
4. What is NodeJs, why do we need it & how does it works
5. What is NPM & how does it works
6. Build tools, it’s working & supported tools
7. Boilerplates & scaffolding
8. Understand folder structure
9. SPA vs SSR
10. What are components structures & how does it work?
11. Atomic design theory
12. What is ReactJS & how does it work?
13. What is JSX & how does it work?
14. Types of components
15. React Lifecycle & it’s methods inside class component
16. States & Props
17. Conditional Rendering
18. Hooks with functional component
19. Networking
20. Third party utilities
21. Styling components
22. Navigation & routing
23. Deploy to Netlify or GitHub Pages
24. Global state management with context api
25. Higher order component (HOC)
26. Error Boundaries

27. Redux with Redux Toolkit
28. Redux Saga
29. Additional Topics
    - Microfrontend
    - Payment gateway integration
    - Firebase
    - Create own feature based NPM packages(Hemant)
    - Architecture level reactjs (A new program)
    - High level software engineer

ReactJS Curriculum Details

1. **Why, what & how?**
    a. Our challenges & blockers
2. **The problem we're trying to solve**
    a. What blocking us
    b. Where we are struggling
3. **Libraries vs Frameworks**
    a. Available players
    b. React vs Angular vs Svelte vs Vue etc
    c. NextJS
4. **What is NodeJs, why do we need it & how does it works**
    a. NodeJS installation & setup
    b. It's working & engine
5. **What is NPM & how does it works**
    a. NPM
    b. NPX
    c. Npmjs.com
    d. Installing a package
    e. Uninstalling a package
    f. Updating a package
    g. Package versioning
6. **Build tools, it's working & supported tools**
    a. Webpack vs ESBuild vs Parcel
    b. Configuration
    c. Minification

    d. Babel
    e. Prettier
    f. ESLint
    g. Configure your own

7. **Boilerplates & scaffolding**
    a. Vite
    b. CRA, etc
8. **Understand folder structure**
    a. Index.html
    b. Package.json
    c. Package-lock.json
    d. Gitignore
    e. Vite.config.js
    f. Src
        i. Main.jsx
        ii. App.jsx
        iii. App.css
    g. ESlintrc
    h. Public
    i. ReadMe.md
9. **SPA vs SSR**
10. **What are components structures & how does it work?**
11. **What is ReactJS & how does it work?**
    a. Virtual DOMs
    b. Diffing Algorithm
    c. React Reconciliation
    d. React Batching
    e. React Fibre
    f. React 18 Architecture change for batching
12. **What is JSX & how does it work?**
    a. JSX
    b. Type Safe
    c. How to write code & allowed syntaxes
    d. Pros & cons
13. **Design System & Atomic design theory**
    a. Components
        i. Atom

        ii. Molecules
        iii. Organisms
        iv. Templates
        v. Pages
    b. Zomato Sushi Design System
    c. Bootstrap
    d. Material UI
    e. Semantic UI
    f. Atlassian Design
    g. Ant Design

14. Types of components

    a. Class Component
        i. Regular Component
        ii. Pure Component
    b. Functional Component
        i. Regular
        ii. Arrow

15. React Lifecycle & it’s methods inside class component

    a. Mounting
    b. Updating
    c. Unmounting

16. States

    a. Getter
    b. Setter
    c. Mutation
    d. Batching
    e. Async
    f. Based on Closure
    g. Callback Function

17. Props

    a. Parent to Child to Sub Child & so on
    b. Child to Parent (Via Callback)

18. Conditional Rendering

    a. Ternary
    b. Short circuit

19. Networking

    a. Fetch

    b. Axios

20. Hooks

    a. Built in Hooks
        i. useState
        ii. useEffect
        iii. useMemo
        iv. useCallback
        v. useReducer
        vi. useRef
        vii. useContext
        viii. useLayoutEffect
        ix. Etc
    b. Custom Hooks

21. Styling

    a. Inline
    b. Class based
    c. Styled Component

22. Third party utilities

    a. React Helmet
    b. Date FNS
    c. MomentJS
    d. Styled component
    e. Dayjs
    f. Mui
    g. React router
    h. Lodash
    i. Axios
    j. Lottie animation
    k. React spinner
    l. Typescript
    m. Formik
    n. React icons
    o. Material icons
    p. React tostify
    q. React suspense layout
    r. React select
    s. React table

t. React pdf
u. React pdf viewer
v. Etc.

23. Navigation & routing
   a. Installing react router dom (Latest)
   b. Configure it & setup routes
   c. Fallback routes
   d. Error handling
   e. Use Link to navigate
   f. Exchange data from one router to another
      i. Query params
      ii. Path params
      iii. State params

24. Deploy to Netlify or GitHub Pages
   a. Setup GitHub pages
   b. Setup Netlify

25. Global state management with context api
   a. Configure Context API
   b. Enable Provider on parent & pass data
   c. Use useContext on child

26. Higher order component (HOC)
   a. For loading
   b. For sharing logic
   c. For handling errors

27. Error Boundaries
   a. Define a HOC to handle error

28. Redux with Redux Toolkit
   a. Install & configure Redux Toolkit
   b. Setup store
   c. Setup provider
   d. Setup middlewares

29. Redux Saga
   a. Setup & connect with redux toolkit
   b. Elements of Redux saga
   c. Define saga file
   d. Connect with store

Milestone to achieve

1. Understand Ecosystem
2. Dive the foundation, tooling system & it’s working
3. Understand components i.e., class or functional
4. Understand lifecycle of a component
5. Play with JSX to write HTML, CSS & JS together
6. Explore & dive state & props to play with data
7. Explore Life Cycle use cases with useEffect & build functionalities
8. Master API calling & play with data & it’s techniques
9. Install & use third party libraries
10. Build apps & beautify them
11. Master navigation system & build pages
12. Build small projects like
    a. Movies Search
    b. GitHub Profiler
    c. Card Game
    d. Tic Tac Toe
    e. Zomato & Swiggy UI with products & searching
    f. Ecommerce with Search, Filters & Cart functionality
    g. New Blogs with comments & like functionality
    h. Splitwise Expense Management
13. Deploy your apps
14. Master Networking with CRUD operation
    a. GET
    b. PUT
    c. POST
    d. DELETE
    e. PATCH

15. Integrate following apps
    a. Firebase
    b. Analytics
    c. Payment (Razorpay)
    d. Sign-In with Google, Facebook, Twitter etc
16. Play with Global State management & scale it
17. Improve your above projects & fine tune them & apply new changes
18. Fine tune & make your app performant with performance optimisation
    a. Explore ReactJS best practises
    b. Apply core web vitals
    c. Compress assets & images
19. Re-Deploy your app
20. Master Redux Toolkit & Redux Saga & apply in your app
21. Test & play with it
22. Re-Deploy it
23. Scale it & keep playing
24. Apply TypeScript & make it even more stronger

<u>Beyond the basics</u>

1. Explore available open source repos
2. Contribute to open source codes
3. Write & read advanced medium articles
4. Explore best practises
5. Explore plugins & articles written by
    - AirBnb
    - Swiggy
    - Flipkart
    - Facebook
    - Microsoft
    - Uber Engineering
    - Etc.

Projects to build

1. Splitwise Expense Management App
2. Social profiler to show all social media account
3. Buffer Clone to schedule social media post
4. Resume builder
5. Kids learning app with graphics & animation
6. Google Forms builder
7. Google Docs builder
8. Draw.io App
9. Whiteboard App
10. VLC media Player
11. Canva Editing Tool
12. CodeShare
13. Google Meet
14. Dynamic UI builder to render entire UI using JSON