React JS Online Course

Course Content

Pre-requisites & Setup

- **Refresh yourself with JavaScript (ES6+), HTML & CSS**
  - let/const, arrow functions, de-structuring, spread/rest, promises, async/await
  - Array & object methods (map, filter, reduce)
- **Tools & environment**
  - Node.js & npm
  - Code editor (VS Code)
  - Git & GitHub basics
  - Browser DevTools

Introduction & Getting Started

- What is React? Why React?
  - Problems it solves (UI updates, re-rendering, componentization)
  - Virtual DOM, reconciliation
- Creating a React app
  - create-react-app (or alternatives e.g. Vite)
  - Project structure & file organization
  - Running, building, and debugging

JSX & Rendering

- JSX syntax & rules
    - Embedding expressions, attributes, children
    - Differences from HTML
    - Conditional rendering, loops in JSX
    - Fragments
- Rendering elements
- “Thinking in React” — breaking UI into components

Components, Props & State

- Components
    - Functional components
    - Class components (for understanding legacy / comparison)
- Props (properties)
    - Passing data into components
    - Default props, prop-types (type checking)
- State
    - Local state in components
    - Updating state, immutability
- Event handling
    - Handling clicks, forms, keyboard events
    - Synthetic events in React

Component Lifecycle, Side Effects & Hooks

- Class component lifecycle methods
    - componentDidMount, componentDidUpdate, componentWillUnmount
- Side effects & data fetching
- Hooks (for functional components)
    - useState, useEffect
    - useRef, useMemo, useCallback
    - useReducer, useContext
- Custom Hooks

Conditional Rendering, Lists & Keys, Forms

- Rendering lists dynamically (using .map)
- Keys & identification in list rendering
- Controlled vs uncontrolled components
- Handling forms (text inputs, radio buttons, checkboxes, selects)
- Validation, error messages

Context & State Management

- Prop drilling problem
- React Context API
    - Creating context, provider, consumer
    - Using useContext
- For larger applications: state management libraries
    - Redux (actions, reducers, store)
    - Redux Toolkit

- Middleware (Redux Thunk, Redux Saga)

Routing & Navigation

- Single Page Applications (SPA) concepts
- React Router (v5, v6)
  - <BrowserRouter>, <Routes>, <Route>
  - Nested routes, route parameters, query strings
  - Link, NavLink, redirecting
  - Handling 404 / NotFound pages

API Integration & Asynchronous Operations

- Fetching data from APIs (REST, GraphQL)
  - fetch / Axios
  - Handling loading, errors
  - CORS, authentication headers
- Async action flows with Redux middleware
- Pagination, filtering, infinite scroll

Performance Optimization & Advanced Patterns

- Code-splitting, lazy loading, React.lazy & Suspense
- Memoization: React.memo, useMemo, useCallback
- Virtualization (for large lists)
- Reconciliation & keys
- Render optimizations, avoiding unnecessary re-renders
- Higher-Order Components (HOCs), Render Props, Compound Components

Testing & Debugging

- Testing React components
  - Jest + React Testing Library
  - Snapshot testing
  - Mocking dependencies
- Debugging techniques & tools
  - React DevTools
  - Browser debugging, breakpoints, console
- Error Boundaries

TypeScript with React

- Why TypeScript?
- Setting up React + TypeScript project
- Typing props, state, hooks
- Interfaces, types, generics in React
- Type-safe context, higher-order components

Server-Side Rendering / Next.js

- Why SSR / SSG?
- Introduction to Next.js
- Pages, routing, data fetching (getStaticProps, getServerSideProps)
- Hybrid rendering
- API routes in Next.js

Deployment & DevOps

- Building for production
- Environment variables
- Hosting / deployment platforms (Netlify, Vercel, GitHub Pages, Heroku)
- CI/CD basics
- Performance monitoring, analytics

Capstone Project

- Build a full-stack app (frontend in React + backend API)
- Real-world features: authentication, CRUD, state management, pagination, file upload, etc.
- Deploy & maintain your app

Total Duration of the Course

- **Total duration:** 2 months.
- **Days:** Monday to Friday (1 hr daily)
- **Mode:** Online

Certificate

- A course completion certificate will be awarded once you have successfully completed this course.

Wishing you a **Smooth Coding** ahead!
