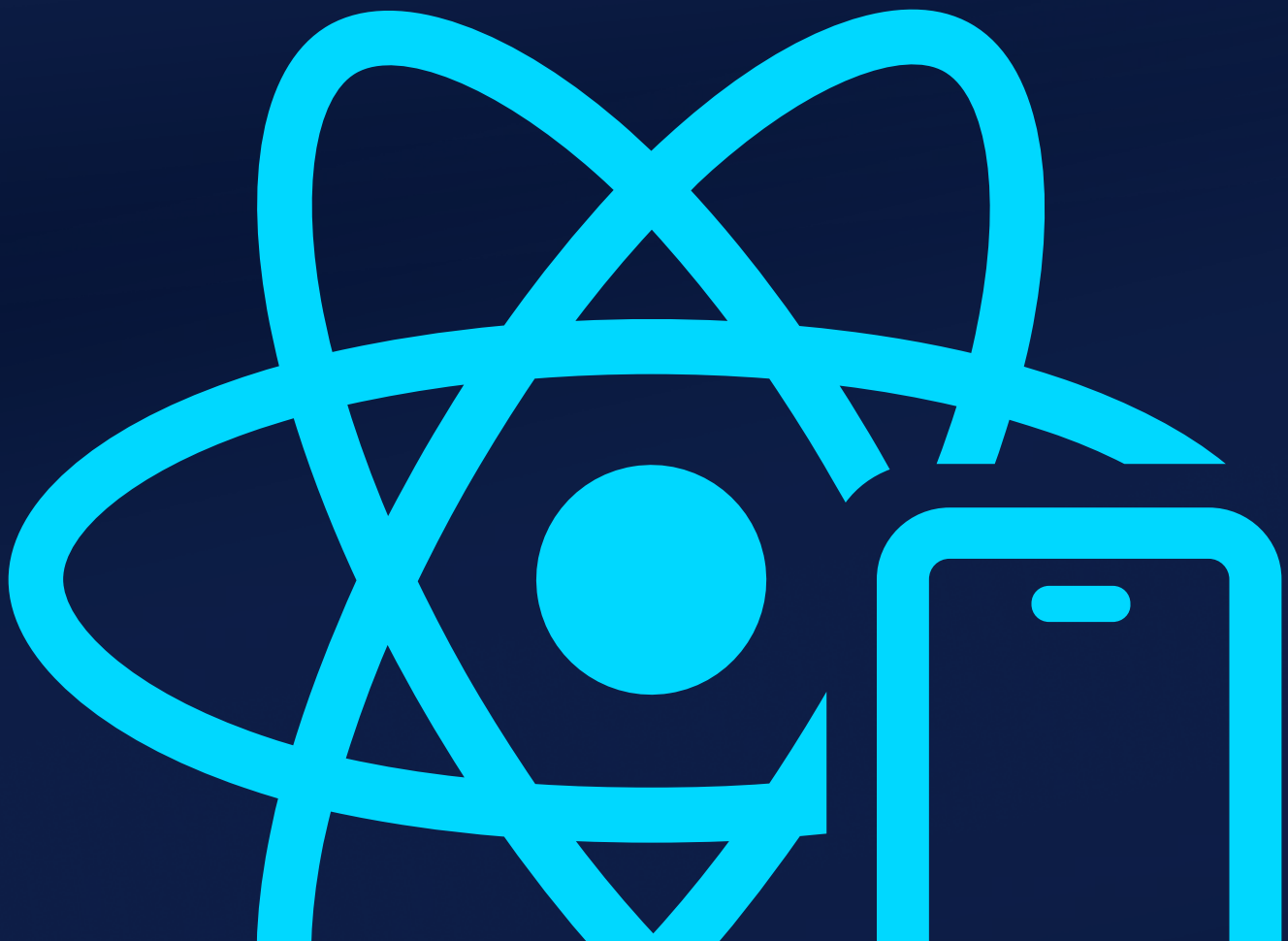
{JS}
MASTERY

The React Native Guide

Created by **JS Mastery**

Visit ***jsmastery.pro*** for more

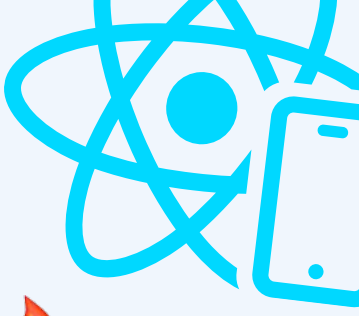


The Ultimate
React Native Course 🔥

After carefully considering your passion for developing full-stack mobile applications from scratch and diving into every aspect of it, we're excited to let you know that your dream of learning and building such apps is within reach.

We're working on a PRO course that will teach you:

- ⭐ Understanding the inner workings of React Native
- ⭐ Mastering React Native fundamentals with a focus on every detail
- ⭐ Creating pixel-perfect UIs for both iOS and Android devices
- ⭐ Developing apps with Expo Go and Emulators
- ⭐ Exploring Expo Router with best practices



The Ultimate
React Native Course 🔥

- ⭐ Fetching APIs using hooks and TanStack Query
- ⭐ Building complex forms with Zod and React Hook Form
- ⭐ Implementing authentication and authorization from scratch (no third-party tools)
- ⭐ Utilizing the latest Expo features, including RSCs and Expo API routes
- ⭐ Creating a custom backend with Node.js and Express.js
- ⭐ Managing databases with MongoDB and Postgres with Prisma
- ⭐ Understanding and using native modules
- ⭐ Animating with React Native Reanimated
- ⭐ Handling payments with Stripe and Razorpay

The Ultimate
React Native Course 🔥

⭐ Conducting unit testing

⭐ Implementing real-time push notifications, messaging, deployment, and more

If this sounds like what you're looking for, join us quickly to stay updated on course details and suggest any features you'd like to see (we value your input).

You can reply to the email from which you received this guide, or drop your suggestion in the

💡| suggestions

Plus, you'll have the opportunity to receive a special discount just for you!

Setup Nativewind

1- Install the library and its dependencies

Native Wind is a library that enables us to apply Tailwind CSS just as we would on a website.

```
npm install nativewind tailwindcss react-native-reanimated react-native-safe-area-context
```

2 - Configure Tailwind.config

Next, run the command npx tailwindcss init to generate the tailwind.config.js file.

```
npm tailwindcss init
```

Here is the default configuration provided:

Setup Nativewind

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: [],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

We need to update the content array to include all the files we want Tailwind to style.

You can do this by adding the file paths/ extensions to the array like this:

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: ["./app/**/*.{js,jsx,ts,tsx}", "./
components/**/*.{js,jsx,ts,tsx}"],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

Setup Nativewind

This means tailwind will affect the app folder and all the files in the app subfolders that have the extensions js, jsx, ts and tsx

Next we have to add NativeWind to the presets, it should look like this:

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: ["./app/**/*.{js,jsx,ts,tsx}", "./
components/**/*.{js,jsx,ts,tsx}"],
  presets: [require("nativewind/preset")],
  theme: {
    extend: {},
  },
  plugins: [],
};
```

Finally create a globals.css file in the app folder and add these Tailwind CSS directives.

Setup Nativewind

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

3- Configure Babel.config.js

To set up the NativeWind preset, follow these steps:

1. Create a New File:

Create a file named babel.config.js in the root directory of your project.

2. Define the NativeWind Preset:

Add the following code to babel.config.js to configure the NativeWind preset:

Setup Nativewind

```
module.exports = function (api) {
  api.cache(true);
  return {
    presets: [
      ["babel-preset-expo", { jsxImportSource:
"nativewind" }],
      "nativewind/babel",
    ],
  };
};
```

4 - Configure Metro.config.js

Run the command npx expo customize **metro.config.js** to generate the **metro.config.js** file in your directory.

```
npx expo customize metro.config.js
```

Now, Access **metro.config.js** and copy the following setup.

Setup Nativewind

This will allow Metro to convert Tailwind CSS styles into React Native styles:

```
const { getDefaultConfig } = require("expo/metro-config");
const { withNativeWind } = require("nativewind/metro");

const config = getDefaultConfig(__dirname);

module.exports = withNativeWind(config, { input: "./app/globals.css" });
```

5 - Import the globals.css file into your root layout app/_layout.tsx

```
import { Stack } from "expo-router";
import "./globals.css";

export default function RootLayout() {
  return <Stack />;
}
```

Setup Nativewind

6 - At the root of the project, create a file named nativewind-env.d.ts & paste the following code inside it:

```
/// <reference types="nativewind/types" />
```

This will enable TypeScript to recognize Tailwind classes, preventing constant error messages.

```
npx expo start --clear
```

7 - Restart your app

At this point, you can start your app and try editing the text or views using Tailwind to verify that everything is set up correctly.

Setup Nativewind

🔴 Make sure to start the app with

```
npx expo start --clear
```

to ensure that the Metro configuration we set up is included in the build process.

8 - Custom themes in tailwind.config.js

We can customize our theme by adding objects to theme.extend in the tailwind.config.js file.

That's it. You're all set.

Introduction

React Native - a framework used to build amazing Android and iOS native mobile applications using the library you already know all about - React.js.

React Native is a cross-platform library that allows you to build native mobile apps using React & JavaScript. This differs from frameworks like Cordova, where you use HTML to create the UI, which will then just be displayed within the device's integrated mobile browser (WebView).

React Native has built-in components compiled to native UI components, while your JavaScript code is executed through a virtual machine. This makes React Native more performant than Cordova.

Prerequisites

Here are the Prerequisites before learning React Native:

1 JavaScript

If you are a complete beginner with no idea about programming, you must first learn JavaScript and understand working with it before learning React or React Native. Without essential concepts of JavaScript, you'll understand nothing.

Start learning the basics of JavaScript at first, and once you are comfortable with that, then move to learn some advanced concepts of JavaScript.

Things you should consider learning →

Prerequisites

- Basic Syntax
- ES6+ features
- Template literals
- Array Methods
- Object property shorthand
- Destructuring
- Rest and Spread operator
- Promises
- Async/Await syntax
- Import and export syntax

Prerequisites

2 Learn React.js

Learn React.js. It’s not officially mandatory to learn React.js before learning React Native but a solid knowledge of React concepts will give you a big advantage in learning React Native.

So I highly recommend learning React first before learning React Native. You don't have to master React.js; you’re good to go with the basics concepts.

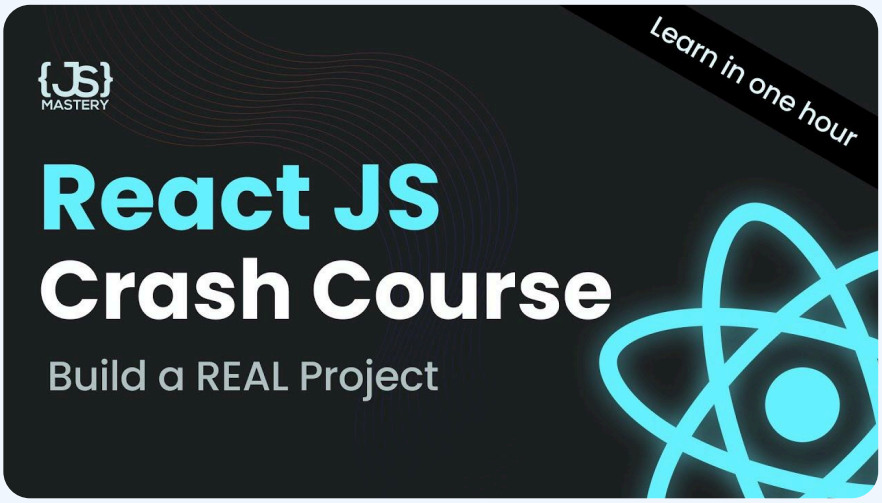
Concepts you should know in React.js →

- File & Folder structure
- Components

Prerequisites

- JSX
- Props
- State
- Styling
- Conditional Rendering

If you haven't learned React yet, I suggest watching my React crash course video.



Video Link

Prerequisites

3 Terminal

You should at least know the primary use of the terminal. In React Native, you'll work with CLI tools such as expo-cli or react-native-cli.

I highly recommend learning how to use the terminal. Also, it will be beneficial for you in the long run.

One more reason to learn terminal is using NPM, NPM does not have a GUI. Every package must be installed manually via the npm command in a terminal window.

This may seem annoying to non-shell users but it actually offers much more control.

Roadmap

Now that you have completed the prerequisites, you can finally start learning React Native.

Environment Setup

You'll have two options to set up a React Native environment.

1. Expo CLI
2. React Native CLI

So which one should you choose? Let's see what React Native recommends.

React Native recommends using the React Native CLI if you are already familiar with Mobile App Development.

Roadmap

However, if you are new to mobile app development and want to get the project quickly set up, Expo CLI is recommended.

Expo is built on top of the react-native CLI and provides many built-in APIs and tools which will take your good time to set up in the react-native CLI.

But why do we have react-native CLI if Expo is that useful? Many native APIs are not yet supported in Expo, like Bluetooth and more. Not all iOS and Android APIs are available. Read more expo limitations here.

Still, I'd strongly suggest that you use Expo as it'll be more than enough for almost all of your projects.

Roadmap

Learn the core components of React Native

Once you set up the environment, you can learn about the core components of React Native. Most apps will end up using one or more of these essential components.

The core components are:

1 View

In React Native, View is a built-in component. If you are familiar with div in HTML, the view is like div; it is used in mobile apps. The view is a content area where you display your content.

Roadmap

2 Text

Text is an essential built-in component; it displays text in the application. The Text component supports nesting, styling, and touch handling.

3 Image

A component for displaying different images, including network images, static resources, temporary local images, and images from local disks, such as the camera roll.

Roadmap

4 TextInput

A foundational component for inputting text into the app via a keyboard. Props provide configurability for several features, such as auto-correction, auto-capitalization, placeholder text, and different keyboard types, such as a numeric keypad.

5 ScrollView

Scrollview renders the extensive list or meaningful content in view with a scrollbar. It helps to view the critical content. It wraps platform ScrollView while providing integration with touch locking "responder" system.

Roadmap

Learn to style your components

1 Style props

You can add styling to your component using style props. You add style props to your element. It accepts an object of properties.

2 StyleSheet

If you have a large codebase or want to set many properties to your elements, writing our styling rules directly inside style props will make our code more complex. For that reason, React Native gives us another way to write concise code using the StyleSheet method.

Roadmap

3 Styled Components

You can use styled-components with React native to write your styles in React Native as you write regular CSS.

Styled-components is a library built for React and React Native developers. It allows you to use component-level styles in your applications.

Styled-components leverage a mixture of JavaScript and CSS using a technique called CSS-in-JS.

Roadmap

Flexbox

React Native uses Flexbox to handle the layout. Flexbox makes it easy to distribute the UI elements in the container. If you don't know it already, then you must learn it. Flexbox is useful in designing your UI and an essential Roadmap component to Learning React Native.

Defining the layout in React Native, it's not that hard; there are a lot of libraries out there that can help you deal with layout, but If you know how to use Flexbox, you won't need to use any libraries.

Roadmap

Navigation

There are multiple screens in almost all the apps; you'll barely find an app with just a single screen. Managing the presentation of screens and transitions between them is handled by what is known as a navigator.

So if you have more than a couple of screens, you need to define routing and navigation that is scalable and easy to maintain.

If you're a beginner, it is recommended to use React Navigation. It's the best navigation library. It provides a straightforward approach to navigation solutions.

Roadmap

Learn Handling Touch

Users interact with mobile apps mainly through touch. They can use a combination of gestures, such as tapping on a button, scrolling a list, or zooming on a map.

React Native provides components to handle all sorts of common gestures and a comprehensive gesture responder system to allow for more advanced gesture recognition.

There are three touchable components. **TouchableOpacity**, **TouchableHighlight**, and **TouchableWithoutFeedback**. Among them, TouchableOpacity is the most widely used one.

Roadmap

Forms

If you're building an app, it's almost a guarantee that you'll have to make at least one form.

You'll probably build sign-ins, sign-ups, and similar. We need a better way of working with forms that are easy to write, maintainable, and, more importantly, developer-friendly.

Developers in the React community build some excellent libraries that make it possible to create complex forms quickly. You can consider the following options:

1. Formik
2. React Hook Form

Roadmap

Learn different ways of debugging

- Learn to debug using console.log
- Learn debugging with Chrome
- Learn debugging in VSCode
- Familiarize yourself with debugging tools

Roadmap

State Management

If the application you're working on is a bigger application, you need a better strategy for managing the state and sharing it across components. To implement a better plan, you'll need to use a state management library.

A state management library is simply a way to engender communication and sharing of data across components. You can consider learning the following options:

1. Redux
2. Mobx

Roadmap

Animations

Animations are essential for a good user experience; in React Native, you have plenty of options to work with energy and work things out of your creativity.

They are mainly used to interact with users' actions, which keeps the user more engaged with your app. Animations can quickly become one of the key factors that users love engaging with on your mobile app.

Some excellent animation libraries are:

- react-native-reanimated
- react-native-motion

Roadmap

Animations

- react-native-animatable
- Popmotion
- React Spring
- React Native Shared Element

Roadmap

Learn to implement Authentication in your app

Almost everywhere, you will need to authenticate users in your application. Authentication allows us to secure our apps or limit access for non-user members.

Authentication can also be used, for example, to restrict access to a paid service or specific service. So it's helpful to learn how to implement Authentication in React Native apps.

Roadmap

Learn Testing

Quality unit testing is super important if you want your application to be more stable. Jest, enzyme, and Detox are great options.

Roadmap

Learn more about different essential components and APIs

- Button
- Switch
- FlatList
- SectionList
- Alert
- Toast
- Modal
- StatusBar

Roadmap

Deployment

You finally made a react native app, and you want your friends and family to be able to use it, or maybe you even want to make a few dollars off it! Publishing your application on the app store and play store takes a bit of time, but I'd still recommend that you learn how to do that.

In the meantime, there's a much better solution. You can have your application online and accessible with everyone, in a matter of minutes! The only thing you need to do is run the command `expo publish`! After running it, you'll terminal with immediately give you the link that you chare share with your friends, and potential employers to get a job.

Roadmap

Optional — Good to know stuff

LESS, SASS

Style your applications in a better way

Typescript

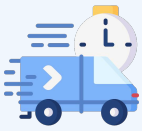
Define your code in a better way

ESLint

Write clean, industry-standard and maintainable code

And much more, there is no end of learning in web/app development there's always something to learn.

Project Ideas



Food Delivery App

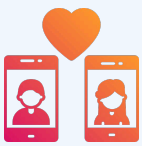
Music App

Chat App

Recipes app

News App

Fitness App



Dating App

Tutor Finder App

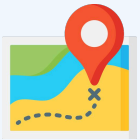
Project Ideas

Job Finder App

Grocery Delivery App

Health Video Chat App

Note-Taking App



Travel-Planning app

Subscription Alert App

Car Sharing App

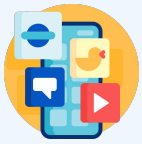
Learning App

Project Ideas

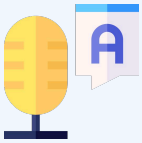
Cryptocurrency Tracker App

Medicine Delivery App

E-commerce App



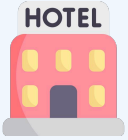
Social Media App



Voice Translation App

Productivity App

Book Review App

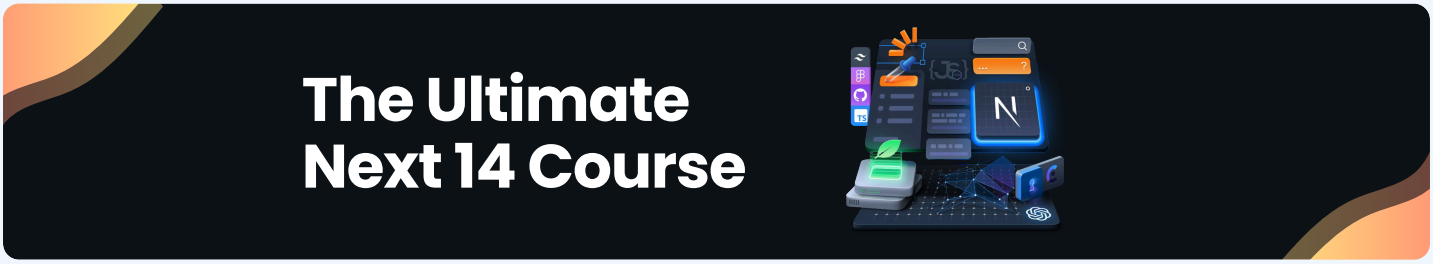


Hotel Booking App

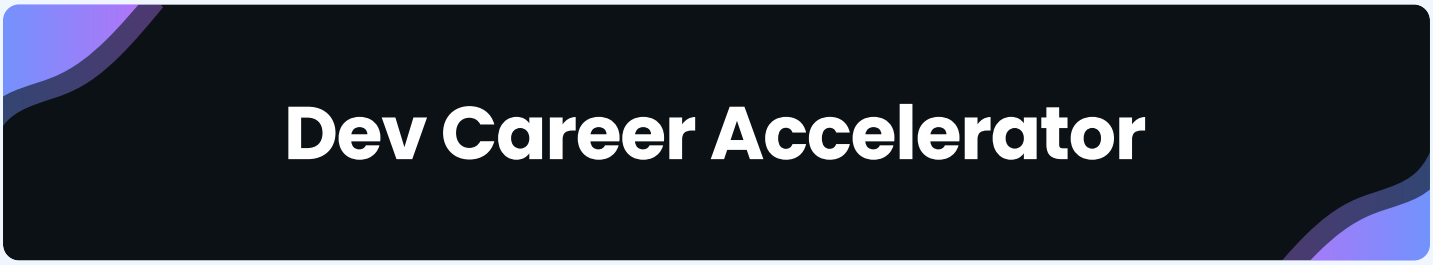
The End

Congratulations on reaching the end of our guide! But hey, learning doesn't have to stop here.

If you're eager to dive deep into something this specific and build substantial projects, our **special course on Next.js** has got you covered.



If you're craving a more personalized learning experience with the guidance of expert mentors, we have something for you — [Dev Career Accelerator](#).



If this sounds like something you need, then don't stop yourself from leveling up your skills from junior to senior.

Keep the learning momentum going. Cheers! 🚀