🔹 React (a.k.a ReactJS)

- **What it is**:
  A **JavaScript library** developed by Facebook (Meta) for building **web applications**.
- **Platform**:
  Runs in the browser (client-side rendering) or with frameworks like **Next.js** (server-side rendering).
- **Use case**:
  - Websites
  - Web dashboards
  - Single Page Applications (SPAs)
- **Tech stack**:
  Uses **HTML, CSS, and JavaScript** (JSX syntax combines HTML + JS).
- **Example**:
  - Facebook web
  - Instagram web
  - Netflix

🔹 React Native

- **What it is**:
  A **framework** built on top of React for building **mobile applications** (iOS & Android).
- **Platform**:
  Runs on **mobile devices**. Instead of HTML/CSS, it uses **native mobile components** (like `<View>`, `<Text>`, `<Image>`).
- **Use case**:
  - Mobile apps
  - Can also target web & desktop via Expo + React Native Web
- **Tech stack**:
  Uses **JavaScript + React concepts**, but styles with **React Native's styling system** (similar to CSS but not exactly the same).
- **Example**:
  - Facebook app

- Instagram app
- Skype
- Tesla app

⚖️ Key Differences

Feature	React (ReactJS) 🖥️	React Native 📱
Platform	Web apps	Mobile apps (iOS + Android)
Components	HTML (div, h1, p)	Native components (View, Text, Image)
Styling	CSS / SCSS	StyleSheet (similar to CSS but limited)
Rendering	DOM in browser	Native APIs for mobile UI
Packages	Web libraries (e.g., React Router)	Mobile libraries (e.g., React Navigation)
Deployment	Web browsers	App stores (Google Play, Apple App Store)

✅ **In short:**

- Use **React** if you're building a **web application**.
- Use **React Native** if you're building a **mobile app**.
- They share the same **React concepts** (components, state, props, hooks), but target **different platforms**.

🔹 What is Expo?

Expo is an **open-source framework and platform** built on top of **React Native** that makes building mobile apps **faster and easier**.

Think of it as a **toolkit + set of services** that helps you avoid the complex setup of native Android/iOS development.

🔑 Key Features of Expo

1. **Zero-config setup** 🛠️
   - No need to install Android Studio or Xcode immediately.
   - You can build and run an app with just `npm install -g expo-cli`.
2. **Expo Go app** 📱
   - You can run your app instantly on your physical device using the **Expo Go app** (just scan a QR code).

3. **Cross-platform support** 🔄
    - Write one codebase → works on **iOS, Android, Web**.
4. **Prebuilt APIs & Components** ⚡
    - Camera, Location, Notifications, Sensors, Image Picker, etc.
    - No need to write native code for these common features.
5. **EAS (Expo Application Services)** ☁️
    - For cloud builds and publishing apps to the App Store & Play Store.
    - Example: `eas build --platform android` to get a production-ready APK/AAB.
6. **Over-the-air (OTA) updates** 🚀
    - Push updates instantly without waiting for App Store/Play Store approval.

📂 Typical Workflow with Expo

Create a new project

```
npx create-expo-app myApp
cd myApp
npx expo start
```

1.
2. **Run on device**
    - Install **Expo Go** from App Store/Play Store.
    - Scan the QR code → app opens instantly.
3. **Develop with Expo APIs**
   Example: Camera access with just a few lines of JS.

Build for production

```
eas build --platform android
eas build --platform ios
```

4.

⚖️ Expo vs Pure React Native

Feature	Expo (Managed Workflow)	Pure React Native
Setup	Easy, no native config	Requires Android Studio/Xcode

Native APIs	Many prebuilt APIs	Add libraries manually
Custom native code	Limited (bare workflow needed)	Full freedom
Updates	OTA updates easy	More manual
Build	EAS build services	Xcode/Gradle builds

✅ **In short:**
Expo = **React Native made easy**.

- Great for beginners and small-to-medium apps.
- For large apps with lots of custom native code → may need to “eject” to **bare workflow** (pure React Native).