

React Native App Development

Course Duration ~65 Hours · 9 Weeks · Live Online · Build-Along · Apps Shipped to Store-Ready Builds

Curriculum reviewed: July 2026 — this ecosystem moves fast and its tutorials lag it; this syllabus is checked against official documentation

New Architecture

Expo & EAS

UI-Thread Animation

Offline-First

OTA Updates

Store Shipping

In short: You already know React. Learn the part that ships to the App Store — native performance, platform behaviour, and releases you can push the same day. Most React Native tutorials still explain the Bridge; it doesn't exist any more: the modern JavaScript interface, renderer and typed native modules are the only architecture that ships, with remaining legacy code being stripped release by release, a new-generation JavaScript engine as the default and precompiled iOS binaries cutting build times. The centre of gravity here is skill reuse and shipping velocity: the managed framework path as the professional default (development builds for full native access, hosted build/submit/OTA services for repeated releases without touching Xcode or Android Studio for most work), over-the-air updates as a product capability within store policy limits, library compatibility auditing as a taught skill, UI-thread animations with worklets, and upgrade planning as routine engineering. React is a prerequisite, not a competitor — if you can build a component and manage state, you know most of this course's language; the mobile part is what this is. The Flutter comparison is fair in both directions: React Native wins where your team already knows React, wants platform-native feel by default, or shares code and people with a web product; Flutter wins on pixel-perfect consistency and breadth of targets; Kotlin Multiplatform where native UX is non-negotiable. Across 9 modules you ship an offline-first app, a native-capability feature, a worklet-driven animation, a performance rescue with profiler evidence, a library audit, an on-device E2E suite, an OTA fix and a store-ready capstone. The AI question, answered honestly: AI writes screens fluently and gets list performance, thread placement, native compatibility and store behaviour wrong — those surface as jank, crashes and rejections, not compile errors.

Duration	9 weeks · ~65 hours · live + hands-on	Format	Live online · build-along · apps shipped to store-ready builds
Prerequisite	Working React and JavaScript. TypeScript strongly recommended and reinforced throughout. No mobile or native development experience required.	Curriculum	9 modules, from the modern architecture to release, operations and capstone
Portfolio	8 production-style builds plus a store-ready capstone	Ideal for	React and web developers moving into mobile, and teams migrating legacy-architecture apps

Why Learn React Native When AI Writes Screens?

AI produces React Native screens fluently and gets mobile-specific things wrong — list performance and re-render scope, animations that run on the JavaScript thread instead of the UI thread, native module compatibility, permissions and platform behaviour, and anything touching the build. Those surface as jank, crashes and store rejections rather than compile errors. The value is architecture, performance judgment and shipping through app review. You'll use AI tooling as an accelerator you review, and learn the failure patterns generated mobile code repeats — so you ship its speed without shipping its mistakes.

Course Highlights

- Every module leads with what you build and ship, not what you "learn about"
- The New Architecture taught as the only architecture — no Bridge folklore
- The managed workflow as the professional default — with the bare escape hatch taught honestly
- Over-the-air updates as a product capability, inside store policy limits
- Animations on the UI thread with worklets — not JS-driven approximations
- Library auditing as a taught skill — assess a dependency before you depend on it
- Upgrade planning as routine engineering — versions leave support quickly here
- Store submission taught properly — the review pitfalls that cause rejections
- AI tooling as an accelerator you review, never a substitute for understanding

Hands-on Projects & Portfolio

- **Multi-Screen App** — file-based navigation, authentication, deep linking and a typed API layer
- **Offline-First Application** — local persistence, background sync and conflict resolution
- **Native-Capability Feature** — camera, notifications and permissions across both platforms
- **UI-Thread Animation Work** — a genuinely smooth interaction driven by worklets, not a JS approximation
- **Performance Rescue** — fix a janky long list and excessive re-renders with profiler evidence
- **Library Audit Exercise** — evaluate real dependencies for architecture compatibility and maintenance risk, and justify the decisions
- **On-Device E2E Suite** — end-to-end tests running on device in CI
- **Over-the-Air Update Workflow** — ship a fix to users without a store submission
- **Capstone** — production app, tested, profiled and submitted as a store-ready build

Tools & Stack

React Native · React · TypeScript · Expo · Expo Router · EAS Build & Update · Hermes · Reanimated · Gesture Handler · TanStack Query · Zustand · MMKV · WatermelonDB or SQLite · Jest · React Native Testing Library · Maestro or Detox · Sentry · Firebase · Fastlane · GitHub Actions · Git

Career Outcomes

React Native's practical advantage is skill and team reuse — a React web developer becomes a mobile developer in weeks rather than months, and one team can maintain web and mobile products sharing language, tooling and often code. That makes it common in startups, product companies and organisations that already have a React web codebase — and the combination of web plus mobile capability in one developer is unusually marketable. This course targets:

- React Native Developer · Mobile App Developer · Cross-Platform Developer · Product Engineer
- Full-Stack Developer (web + mobile) · Freelance or agency app developer · Technical founder shipping a mobile product

Who This Course Is For

- **React and web developers moving into mobile** — the primary audience
- Final-year students and fresh IT graduates targeting mobile or product engineering roles
- Full-stack developers who need to ship a mobile front end for their own product
- **Teams maintaining older React Native apps still on the legacy architecture** — the migration is now mandatory rather than optional
- Native Android or iOS developers adding cross-platform reach
- Freelancers and founders shipping to both stores with limited resources

Module 1: How React Native Actually Works

8 Hours

YOU SHIP

- Your first running app — plus an annotated map of the modern architecture, tracing one interaction from JavaScript through the renderer to native and back

TOPICS

- The modern architecture: the JavaScript interface layer, the current renderer, typed native modules and code generation
- The JavaScript engine and ahead-of-time bytecode; the threading model
- What replaced the Bridge — because everything you debug later depends on understanding this

Assessment: Quiz + Lab

Module 2: Project Setup & Tooling

7 Hours

YOU SHIP

- A TypeScript project on the managed workflow with a development build giving full native access — plus a practice run of the upgrade workflow, treated as the routine task it now is

TOPICS

- The managed framework workflow and development builds; the bare workflow and when it's justified
- TypeScript configuration; native project structure without fear; environment and configuration management
- Debugging tools; the upgrade workflow as a routine task

Assessment: Quiz + Lab**Module 3: Building Mobile Interfaces**

7 Hours

YOU SHIP

- A platform-adaptive, themed, accessible interface — respecting iOS and Android conventions, safe areas, dark mode and tablet layouts — from one codebase

TOPICS

- Core components and the styling model; flexbox on mobile; platform-adaptive design for iOS and Android conventions
- Safe areas; responsive and tablet layouts; dark mode and theming
- Gesture handling; accessibility as a build requirement

Assessment: Quiz + Lab**Module 4: Navigation & App Structure**

7 Hours

YOU SHIP

- A multi-screen app with file-based navigation, deep links, authenticated route guards and state that survives relaunch — structured to survive a growing team

TOPICS

- File-based and stack/tab/drawer navigation; deep links and universal links; authenticated route guards
- Modal and nested navigators; state persistence across launches
- Project structure that survives a growing team

Assessment: Quiz + Lab**Module 5: State, Data & Offline**

8 Hours

YOU SHIP

- An offline-first application: typed API layer with caching, secure credential storage, local persistence, background sync and conflict resolution — mobile requirements, not optional extras

TOPICS

- Component state versus server state; data fetching and caching; typed API layers
- Secure credential storage; local persistence; offline-first behaviour and sync conflict handling
- Background refresh; real-time updates

Assessment: Quiz + Lab**Module 6: Native Capability & Platform Integration**

8 Hours

YOU SHIP

- A native-capability feature — camera, push notifications and permissions handled across both stores' privacy rules — plus the library audit exercise and a simple native module written from scratch

TOPICS

- Camera, location, notifications and push; permissions and the privacy prompts each store enforces; biometrics; file and media handling; background tasks; in-app purchases
- Auditing third-party libraries for architecture compatibility before you depend on them
- Writing a simple native module when nothing suitable exists

Assessment: Quiz + Lab**Module 7: Performance**

8 Hours

YOU SHIP

- The performance rescue: a janky long list and excessive re-renders fixed with profiler evidence — plus a genuinely smooth UI-thread animation driven by worklets

TOPICS

- List virtualisation and why long lists are where most apps fall over; re-render scope and memoisation
- Animations that run on the UI thread with worklets; image handling
- Bundle size and startup time; memory profiling; diagnosing jank with profiler evidence rather than guesswork

Assessment: Quiz + Lab**Module 8: Testing & Quality**

6 Hours

YOU SHIP

- A fully tested feature — unit and component coverage, mocked native modules, and an end-to-end suite running on device in CI — with crash-free sessions tracked

TOPICS

- Unit testing; component testing; end-to-end testing on real devices and simulators; mocking native modules
- Snapshot and visual regression testing; crash-free session tracking
- Static analysis; continuous integration for a mobile project

Assessment: Quiz + Lab**Module 9: Release, Operations & Capstone**

6 Hours

YOU SHIP

- An over-the-air fix pushed to users without a store submission — and your capstone: one production-grade app, tested, profiled and submitted as a store-ready build through a hosted pipeline

TOPICS

- Hosted build and submission pipelines; signing and provisioning without the usual pain; build profiles for development, staging and production
- App Store and Play Store review pitfalls that cause rejections; staged rollouts; over-the-air updates and their policy limits
- Crash reporting, analytics and release monitoring; capstone design, build, test and ship

Assessment: Capstone Project + Store-Ready Build