How I built a production-ready full stack app in 1 day at almost no cost: React Native, Firebase, Cloudflare

clara · Follow
Published in Women in Technology · 5 min read · 3 days ago

Image by ThisisEngineering from Unsplash

Table of contents:

1. My goal
2. The tech stack that made it happen
3. Costs
4. A pesky bug to avoid

My goal

Before diving into the “how”, let me take a step back and share my goal.

I wanted to launch quickly, experiment with ideas and get users, so the app needed to be production-ready right out of the gate. The stack had to be fast, secure, simple, and cost-effective.

The tech stack that made it happen

1. React Native

For someone coming from a web dev background, React Native is a no-brainer. If you’re familiar with ReactJS, you can jump right in without needing to learn a whole new framework or language. Plus, you get the added bonus of building for both iOS and Android with a single codebase. A great way to hit the ground running.

Protip: Use the Expo framework to abstract away some low level complexity and allows you to build even faster.

2. Firebase

It was my first time using Firebase, and honestly, it was a breeze. Firebase offers an all-in-one managed backend that takes care of the heavy lifting for you.

With minimal setup, I was able to integrate things like authentication for logins (with various auth providers), storage (for images/documents), cloud functions (for server side logic) and hosting.

Authentication

This is probably my favourite feature. Firebase's authentication SDK is incredibly flexible and easy to use. Here are some key options:

- Email/password — Firebase handles password storage and authentication flow so you don't need to.
- Social sign ins — integrate with providers like Google, Apple, Facebook to improve user experience.
- Anonmyous authentication — this lets users explore the app without creating an account

I've linked the auth docs here.

Firebase App Check

App Check is a security feature that protects your backend from abuse by verifying that incoming requests originate from your app. This is super handy if you have a custom backend hosted elsewhere.

Firebase hosting

To deploy your codebase to Firebase hosting, here are the steps:

- Install Firebase CLI: `npm install -g firebase-tools`
- Login to Firebase and authenticate: `firebase login`
- Initialize Firebase hosting in your project's directory: `firebase init hosting` . You will be asked to specify the public directory where your compiled files are located. (Mine was the `dist` folder).
- Build the project.
  If you're using React Native with Expo, use: `npx expo export` This builds files across all 3 platforms (web, android, iOS).

- Deploy: `firebase deploy --only hosting` (this flag deploys just the hosting configuration and not other Firebase features)
- That's it! Now head over to the console and you will see the URL of the app.

Custom domains

If you want to make app more professional, it's probably worth purchasing a **custom domain** from providers like GoDaddy or Namecheap and link it to Firebase hosting.

The process is straightforward: Go to the Firebase Console, navigate to Hosting → Connect Domain, and follow the instructions to update your DNS settings.

Firebase automatically generates SSL/TLS certs for your domain. This ensures that all HTTP requests are converted into encrypted HTTPS.

3. Cloudflare

Security is a non negotiable, even when moving fast.

While Firebase automatically generates SSL/TLS for your site (including custom domains), it lacks some critical features:

- A **WAF** (web application firewall) to protect against vulnerabilities like SQL injection and cross-site scripting.
- Bot detection and mitigation to filter out spam and malicious bots.
- Global DDoS protection to block attacks before they reach Firebase's infrastructure.

The best part? Cloudflare's **free tier** is super generous and doesn't require a credit card to sign up :) I didn't have to upgrade my account!

Protip: Do configure email alerts to be notified of potential threats.

4. Bonus: Google's reCAPTCHA v3

I would highly recommend this free service, which runs invisibly in the background and assigns a confidence score (0.0 to 1.0) to interactions, letting you detect whether it's likely a human or a bot.

A quick rundown on how I did it:

- First, I added reCAPTCHA to key parts of my app, like signups and forms. This involves writing a function on the frontend that uses the reCAPTCHA script to generate a token whenever a user performs an action.
- The token generated by reCAPTCHA **should not be validated on the client-side** (that's insecure!). Instead, the token is sent to the backend for verification. I used Firebase's cloud functions for this step, which communicates with Google's reCAPTCHA API to verify the authenticity of the token.
- Once the token is validated, a confidence score is returned, and you can then decide if you'll let the user through.

Costs

So how much did everything cost me?

Aside from purchasing a custom domain name (cost varies depending on the domain you pick), the only thing I upgraded from a free tier was Firebase. I wanted to ensure my app wouldn't fail if I hit the free tier limits.

Firebase offers two pricing plans:

1. **Spark Plan (no cost)**: Great for small apps or testing, but with limits on storage, hosting, and usage.

2. **Blaze Plan (pay-as-you-go)**: Scales with your app's needs. You're only charged for what you use beyond the free tier limits.

Frankly, I do think the free tier is pretty generous. For instance:

- Cloud Firestore: 50,000 free reads and 20,000 free writes per day.
- Cloud Functions: 2 million free invocations per month.

Pricing docs here.

I would also recommend you to set up billing alerts. You can also downgrade back to the Spark plan or upgrade further depending on your needs.

A pesky bug to avoid

When using Firebase hosting and Cloudflare together, you might encounter a common issue: the dreaded **"Too Many Redirects"** error. Novice me spent a good hour troubleshooting this, so hope this saves you some time!

What went wrong:

1. When a user visits the site via HTTP, Firebase automatically redirects them to HTTPS.
2. Simultaneously, Cloudflare also detects HTTP traffic and attempts to redirect it to HTTPS.
3. The double redirection causes an endless loop, resulting in the "Too Many Redirects" error.

How to fix it:

1. Firebase automatically directs HTTP traffic to HTTPS and so we want to disable this feature in Cloudflare to let Firebase handle it exclusively.
   Go to the Cloudflare dashboard -> SSL/TLS -> Edge Certificates -> disable "Automatic HTTPS Rewrites".

2. Remember to configure SSL/TLS encryption to Full Strict mode.
   Full Strict ensures that Cloudflare accepts only **valid SSL certificates** from your origin server (Firebase) that are signed by a trusted Certificate Authority, like Let's Encrypt.

To end off

That's all from me! I hope these tips are helpful for startups looking to prototype quickly or for anyone building side projects and trying to attract users.

And if you think that building a full stack app for production is daunting and potentially costly, I hope I've changed your mind.

Feel free to drop a comment if you have questions or feedback, or connect with me here :)

Have fun shipping code!

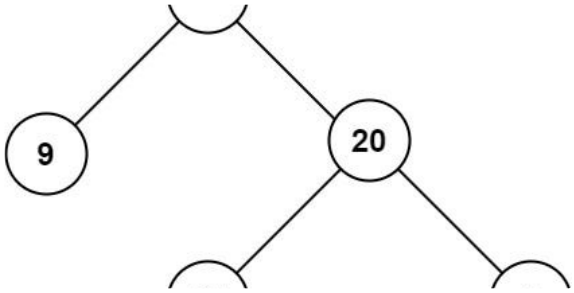
Women In Tech | Firebase | Cloudflare | React Native

No responses yet

What are your thoughts?

Respond

More from clara and Women in Technology



clara

Building intuition of Depth First Search (DFS) using the call stack

If you're a software engineer and had to ever practice for any coding interviews, chances...

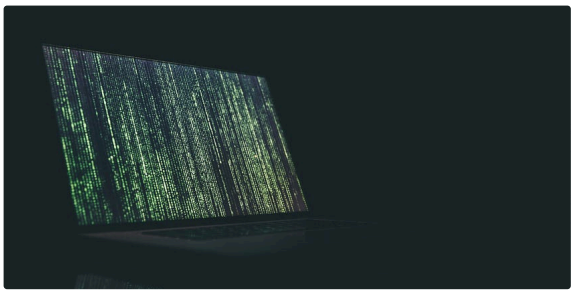
Aug 15 78

In Women in Techn... by Angi Milano @ Maven Ad...

How to Earn More and Negotiate with Confidence

Negotiating with Confidence: Lessons Learned and Strategies to Advocate for...

Nov 26 117



In Women in Technology by Avishi Gupta

Shoot Your Shot: My No Plan Promotion Journey At Google

I have no plan of being modest in this article.

Nov 9 156 1

In Women in Technology by Beatrice Murage

Job Transition in Tech: How to Make a Strategic Move Without Losing...

Transitioning from one job to another can be both exciting and daunting, especially in the...

Nov 14 59

See all from clara

See all from Women in Technology

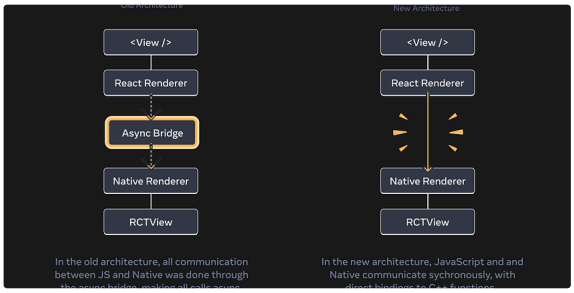
Recommended from Medium

Ham Na Nakorn

Building a Real-Time ChatGPT App with Next.js, TypeScript, and...

In this blog, I'll walk you through building a simple, real-time chat application using...

Oct 11 4



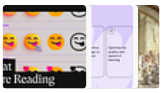
In JavaScript in Plain Engl... by Teenage Program...

React Native Just Got 550% Faster

React Native's new architecture is here! Get no delays and extreme app performance! 🚀

Oct 28 158 8

Lists



Stories to Help You Grow as a Designer

11 stories · 1046 saves

Leadership

62 stories · 495 saves

Stories to Help You Grow as a Software Developer

19 stories · 1503 saves

Best of The Writing Cooperative

67 stories · 449 saves

The Expert Developer

Step-by-Step Guide: Setting Up Splash Screens in React Native fo...

A splash screen is an introductory screen

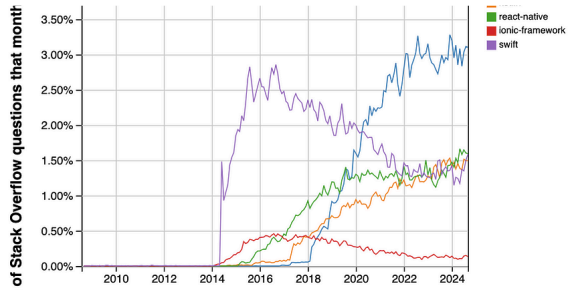
DevNex Pro

Handling API Calls Gracefully in React Native: Best Practices for...

React Native has become a popular

Open in app ↗

Medium Search Write



In Level Up Coding by vvs

Flutter ain't going away: Stackoverflow has spoken

There are rumours and articles suggesting that Flutter may be on its way out....

Nov 18 316 2

Gautham Vijayan

React Native 76.0—Exploring & Enabling new architecture by fixin...

And boost your app's performance!

Oct 27 43

See more recommendations