



Creating a React Native App with Node.js backend — (Part 1)



Oluwakorede Fashokun · [Follow](#)

Published in RedCoder · 3 min read · Apr 2, 2018



659



10



This is part one of a three-part series on creating a React Native Application, with a Node.js API behind the hood. Basically, this is the first step in creating a dynamic mobile application; which works using Javascript on both the frontend and the backend. “How is this done?” you ask. We will cover how this works. This article assumes that you know what Node.js is and can do. If you don’t, bookmark this post, watch one of Brad Traversy’s Node.js YouTube videos and come back later.

Note: To complete this tutorial successfully, you will need to have Node.js installed on your machine. Simply visit this [link](#), and follow the instructions provided. All code used in this tutorial are available in [this](#) GitHub repo.

What we will be building:

We will build a simple mobile application, that views random posts from our API. We will be using the following the following tools:

- Your favourite text editor (I use Vim, but I recommend VSCode).
- An API management tool (Postman or Insomnia - I'll use Insomnia)
- A command terminal (Bash for Linux and MacOS, Command Prompt, PowerShell or Git Bash for Windows)
- Spare 45 minutes.

What is React Native and why are we using it?

React Native is simply a framework built by the Facebook developer team (and many others, as its open source), use to write truly native apps in JavaScript. You're thinking, "But Cordova and Ionic already do that!". Cordova and Ionic build "hybrid" mobile applications - wrapped in a web view (basically a website running out of the web browser, but with a native wrapper). But React Native uses a "bridge" to translate all your JavaScript code to the target device's native language (Java on Android and Objective-C on iOS), making it completely native, while allowing you to port your React and JavaScript knowledge from the web. This effectively allows web developers to easily create truly native applications. That sound good? There's more. You can also use your CSS skills to style your React Native

elements, as Facebook's Devs implemented CSS-like code for styling. To learn more about React Native, read the docs at <https://reactnative.dev> (and thank me later!).

. . .

Getting Started

Create a new folder on your computer and create a sub-directory, named backend.

Open your favourite command terminal inside the **backend** folder and run the following commands:

```
yarn add express mongoose body-parser connect-flash dotenv express-validator  
yarn global add expo-cli
```

If you did the preceding correctly, you should find a `node_modules` directory with some packages inside it. Including `express`, `mongoose`, `body-parser` and `dotenv`. Let us quickly see what these packages do.

express - This is a very popular (if not the most popular) node package. Express makes creating a Node server less cumbersome and easier to accomplish. Simply put, it makes using Node easier.

mongoose - As we will be using, MongoDB as our database, we need a middleware that helps us handle database connections, table and documents creation, edition and deletion.

body-parser - We'll use this to parse form inputs, so that express can pass them on to the database.

dotenv - This is used to manage environment variables.

express-validator - This is used to validate form entries (verify email authenticity, password match etc.)

Now, create a new terminal window inside the **project** folder (not the **backend** folder), and run the following commands:

```
expo init posts-app
```

This will create a directory named **posts-app**. Now we will need to rename this directory to **client** (for the sake of simplicity). If you use a MacOS, Linux or Git Bash on Windows, simply run:

```
mv posts-app client
```

Note: You must run this code in the project directory (and not in the backend sub-directory)

Congratulations if you made it through with out getting terminal-burnt. In part 2, we will dig into some real code. You can also find the code in [this GitHub repo](#). See you in part 2!

React Native

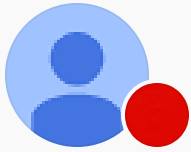
JavaScript

Nodejs

Expo

Mobile App Development

 Unlisted



Written by Oluwakorede Fashokun

114 Followers · Editor for RedCoder

Follow



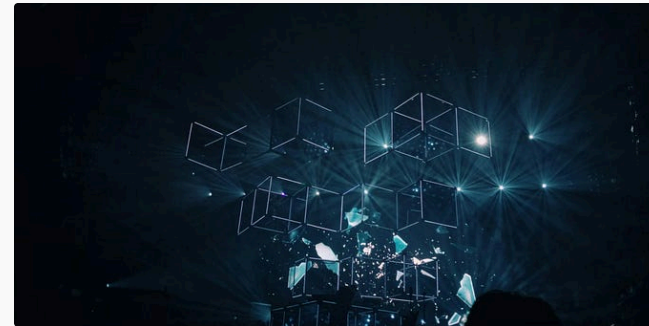
Recommended from Medium



Jamon Holmgren in Red Shift

Microsoft is retiring App Center: Here's what React Native...

Big bummer, React Native developers:
Microsoft announced they are shutting dow...



Onix React

Release React Native 0.74.0

React Native 0.74 has been released, bringing
with it a slew of updates designed to enhanc...

8 min read · Apr 23, 2024

 886  2



4 min read · Apr 23, 2024

 578 



Lists



Stories to Help You Grow as a Software Developer

19 stories · 1027 saves



General Coding Knowledge

20 stories · 1183 saves



data science and AI


40 stories · 144 saves



Generative AI Recommended Reading


52 stories · 990 saves



 Oshen Dikkumbura in Stackademic

React Native Navigation With TypeScript



 Matheshyogeswaran

Building Authentication in React Native with Firebase: A Step-by-...

Navigation is a key feature in today's mobile apps. When it comes to the React Native...

6 min read · Nov 8, 2023



```
const data: {
  id: string;
  image: string;
  price: number;
}[] = await response.json();
setListProduct(data);
} catch (error) {
  console.error('Error fetching data: ', error);
}
};

getProductList();
}, []);

return (
  <View>
```



Charisma Kurniawan Aji

Make your React Native code cleaner

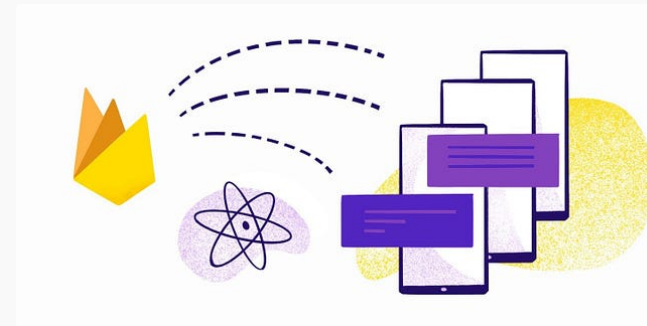
Feeling overwhelmed by a cluttered mess of React Native code? As your app expands, so...

3 min read · Mar 8, 2024



Here, we are going to delve into setting up React Native with Firebase and then setting...

4 min read · Feb 22, 2024



Abbas Ali Radiowala

React Native Push Notifications: Your Ultimate Guide

Hi Geeks !! In this fast-paced world of mobile app development, Push Notifications play a...

9 min read · Feb 13, 2024



See more recommendations

