

race.jsx is loading...



Check out:

[react-cdn-firebase-post-app](#) ([code](#))

STORAGE & NATIVE APIS



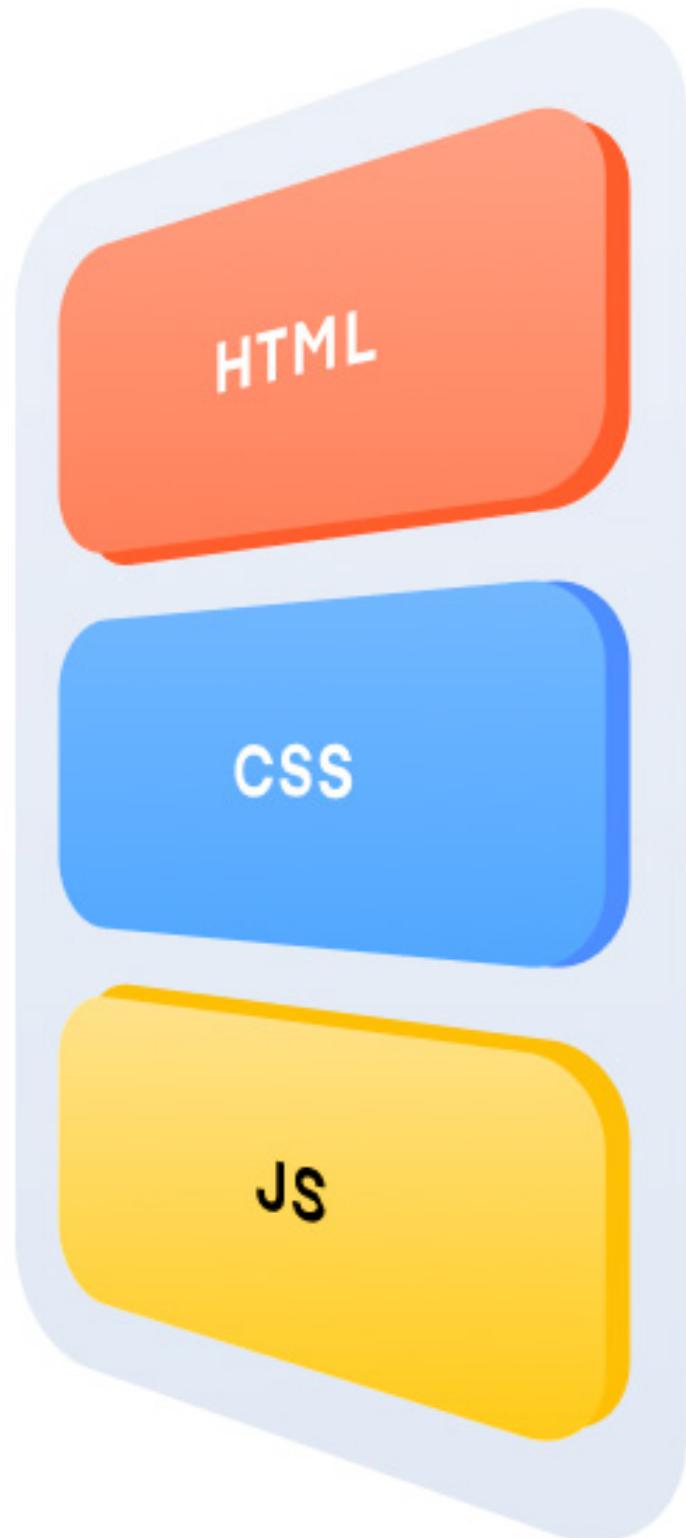
Agenda

APP Development

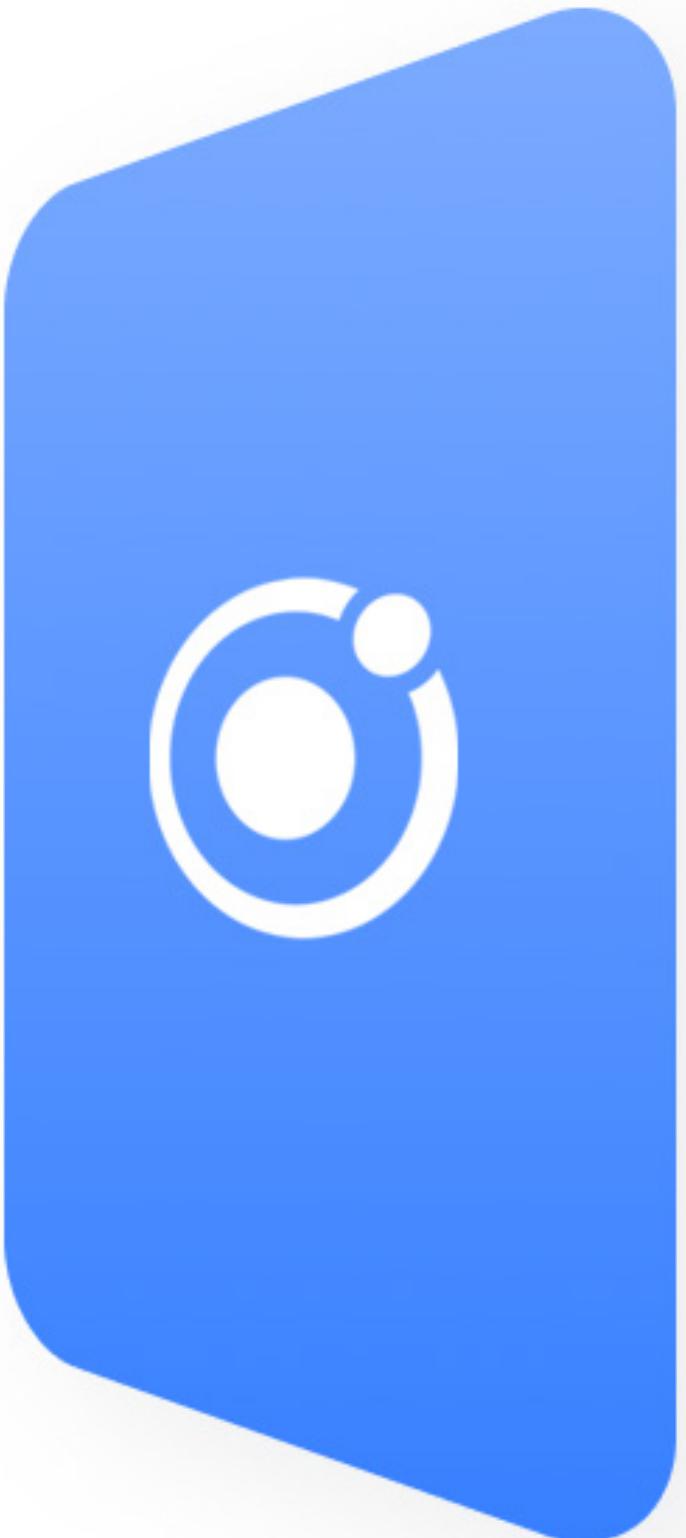
Ionic Post App w/

- Firebase REST
-
-
-
-

1. Deployment 🧑
2. Ionic Post App w/ Firebase REST
 - Firebase & Realtime DB REST API
 - Native APIs
 - Icons & Splash Screens
3. Internship Info (13.15)



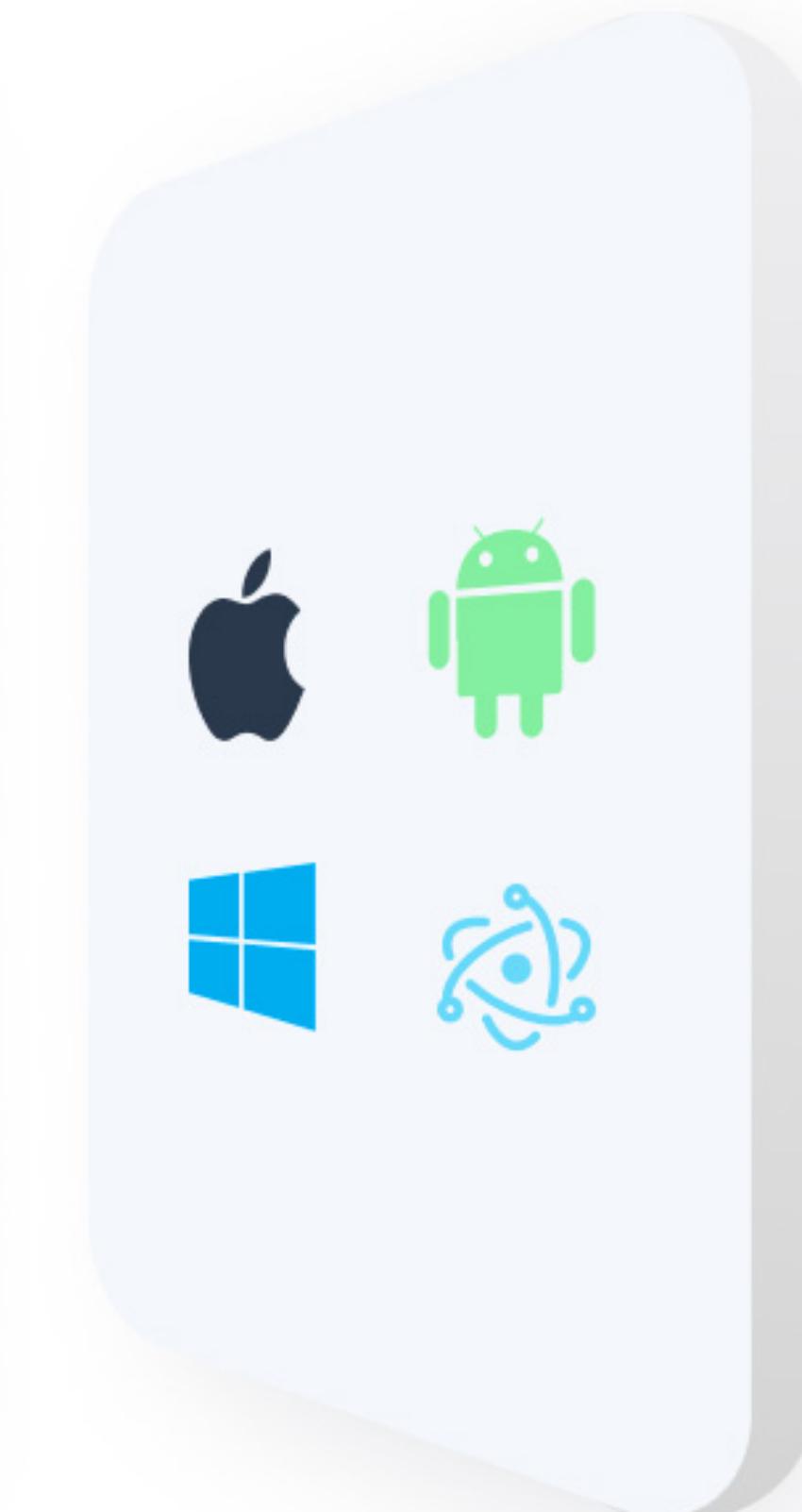
YOUR APP (ANGULAR, REACT, VUE...)



UI CONTROLS (IONIC)

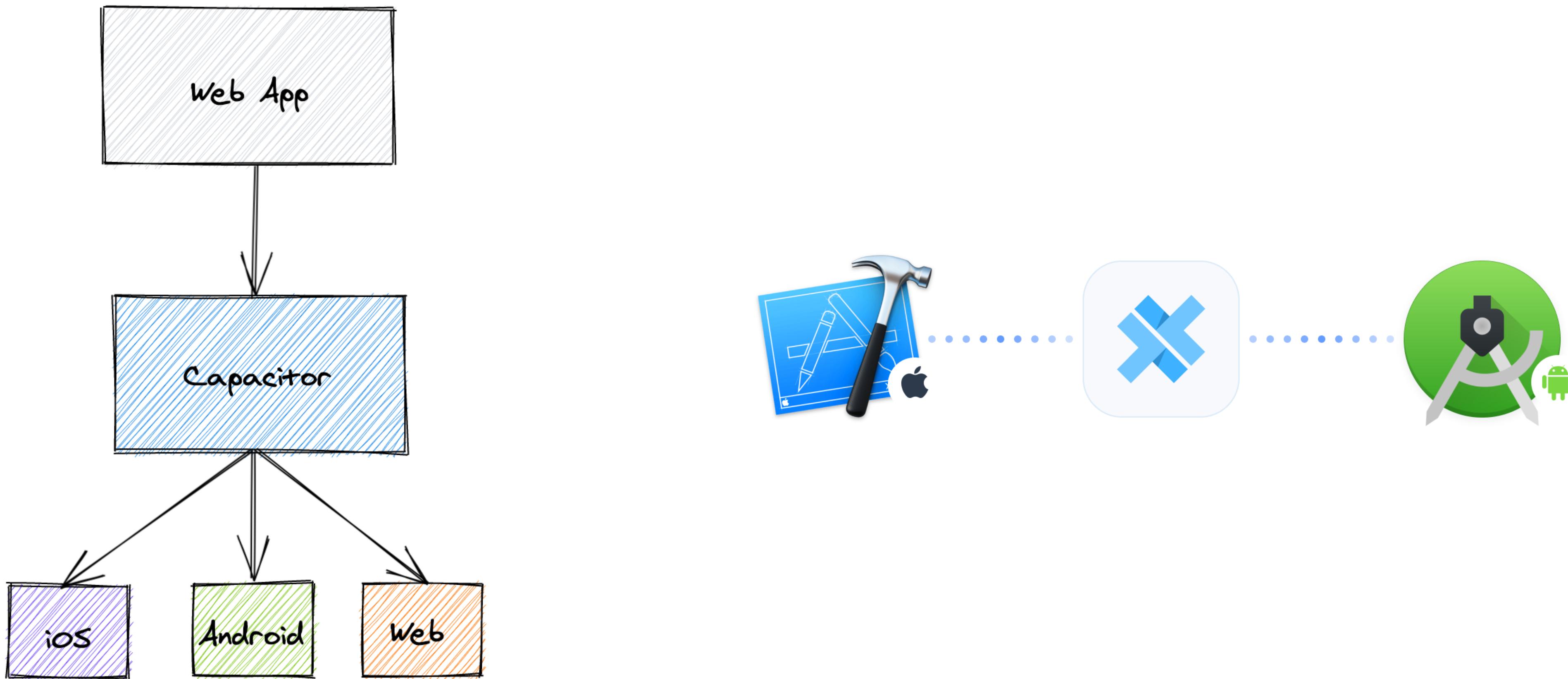


NATIVE ACCESS (CAPACITOR)



DISTRIBUTION PLATFORMS

App Dev w/ Capacitor





App Dev can be tough 🙃

You develop 90% in the
browser

... and then you test on devices (physical devices!)

Deployment



1. \$ ionic build
2. \$ ionic cap add [platform]
3. \$ ionic cap copy or \$ ionic cap sync (plugins)
4. \$ ionic cap open [platform]
5. \$ ionic cap run [platform]
6. \$ ionic cap run [platform] -l --external (Livereload)

Deploying to iOS and Android

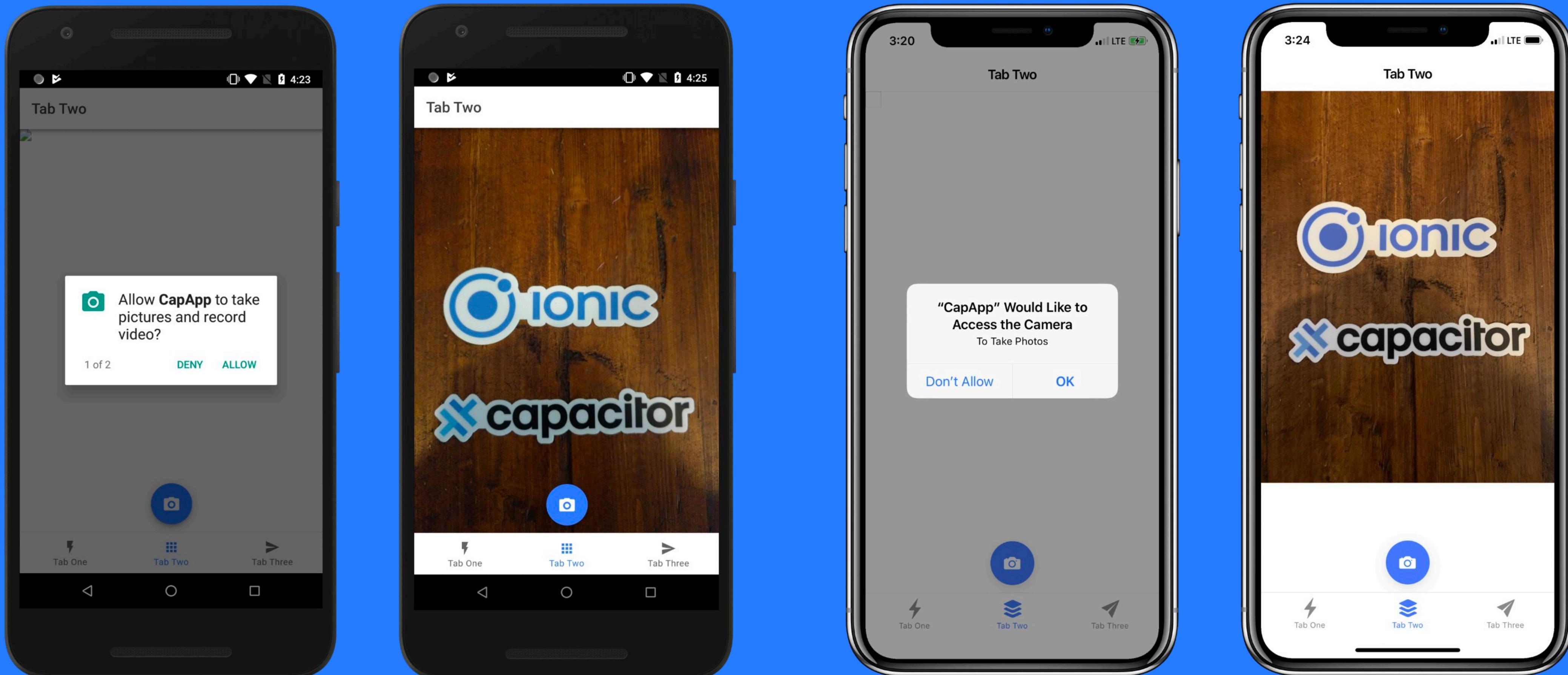
Android & Livereload



1. Update Java Version
2. Android Studio -> SDK Manager -> Android 9.0(Pie)
3. Turn on developer mode on your phone, Allow USB Debugging and connect your device via USB.
4. Through Android Studio, choose your physical device and run project.
5. Run `$ ionic capacitor run android -l --external` and choose your device.

User Permissions

<https://ionicframework.com/docs/react/your-first-app/deploying-mobile>



Configuring Android

<https://capacitorjs.com/docs/android/configuration>

The screenshot shows a web browser window with the title 'Configuring Android - Capacitor'. The URL in the address bar is <https://capacitorjs.com/docs/android/configuration>. The page content is titled 'Setting Permissions'. It explains that permissions are defined in `AndroidManifest.xml` inside the `<manifest>` tag. An example code snippet shows how to add Network permissions:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.getcapacitor.myapp">  
    <activity>  
        <!-- other stuff -->  
    </activity>  
  
    <!-- More stuff -->  
  
    <!-- Your permissions -->  
  
    <!-- Network API -->  
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />  
</manifest>
```

At the bottom, a note states: 'Generally, the plugin you choose to use will ask you to set a permission. Add it in this file.'

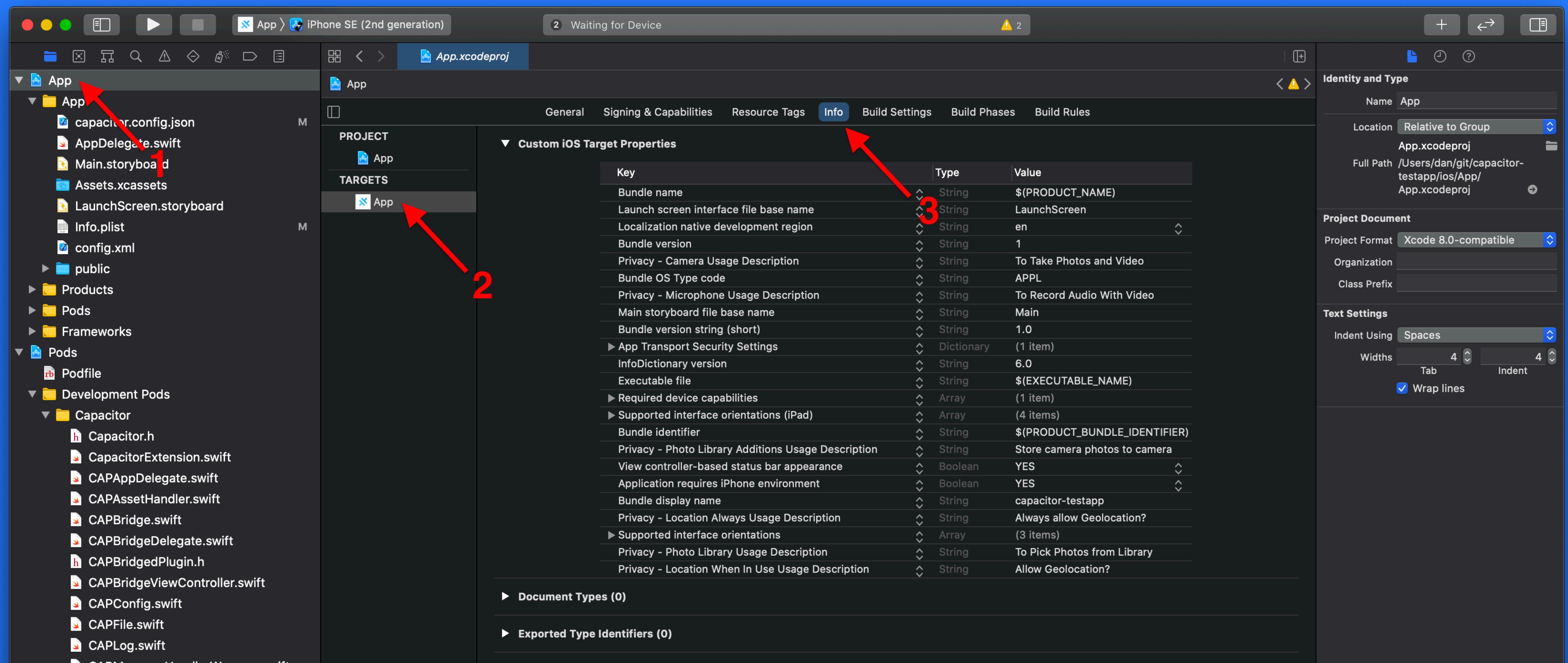
The left sidebar contains navigation links for 'Environment Setup', 'Installation', 'Using with Ionic Framework', 'Basics' (with sub-links for Development Workflow, Using Plugins, Native Project Configuration, and Utilities), 'Upgrade Guides', 'Cordova/PhoneGap', 'Concepts', and 'iOS' (with sub-links for Getting Started, Configuration, Custom Native Code, Deploying to App Store, Custom ViewController, and Troubleshooting).

The top right features a search bar, navigation links for 'Docs', 'Plugins', 'CLI', 'Community', 'Blog', and 'Enterprise', and social media icons for GitHub, LinkedIn, and Twitter. A blue 'Install' button is also present.

The right side of the page includes a 'CONTENTS' sidebar with links to 'Configuring AndroidManifest.xml', 'Changing the Package ID', 'Changing the App Name', 'Deeplinks (aka Android App Links)', 'URL Schemes', and 'Setting Permissions'. Below this is a 'Submit an edit' button and an 'appflow' logo with the tagline 'Continuous app delivery made easy. Build, publish, and update from the cloud.'

Configuring iOS

<https://capacitorjs.com/docs/ios/configuration>



Common Commands

<https://race.notion.site/Ionic-CLI-Common-Commands-7715d0d0c3754bbc97bd3f0a47ddb975>

Dev & deployment tips

<https://ionicframework.com/docs/developing/tips>

<https://ionicframework.com/docs/cli/livereload#tips>

<https://ionicframework.com/docs/react/your-first-app/deploying-mobile>

Ionic Post App w/ Firebase REST

Step 1 - 10

Firebase & Realtime DB REST API

Step 1 - 5

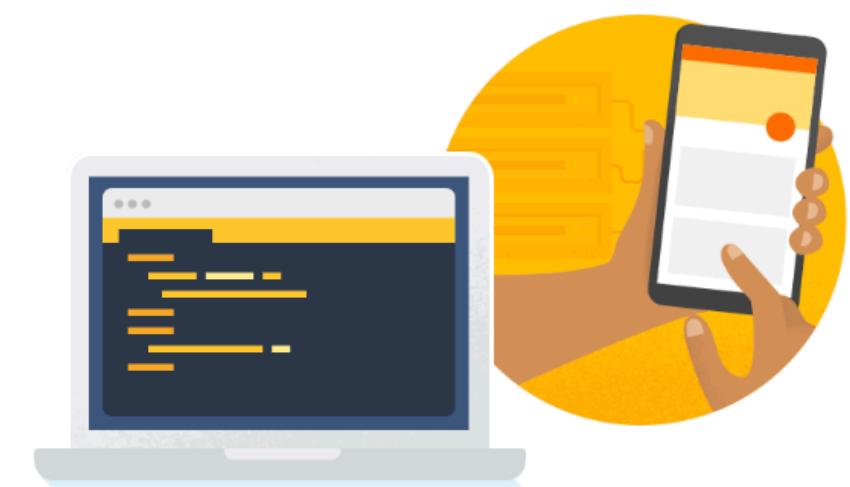


Introducing Firebase



What is Firebase?

Platform & Backend-as-a-Service
for Web & App Development



Build better apps



Cloud Firestore

Store and sync app data at global scale



ML Kit BETA

Machine learning for mobile developers



Cloud Functions

Run mobile backend code without managing servers



Authentication

Authenticate users simply and securely



Hosting

Deliver web app assets with speed and security



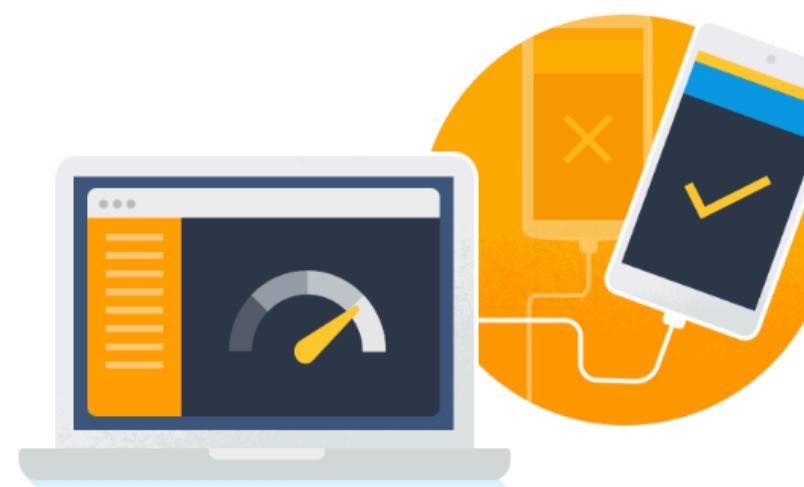
Cloud Storage

Store and serve files at Google scale



Realtime Database

Store and sync app data in milliseconds



Improve app quality



Crashlytics

Prioritize and fix issues with powerful, realtime crash reporting



Performance Monitoring

Gain insight into your app's performance



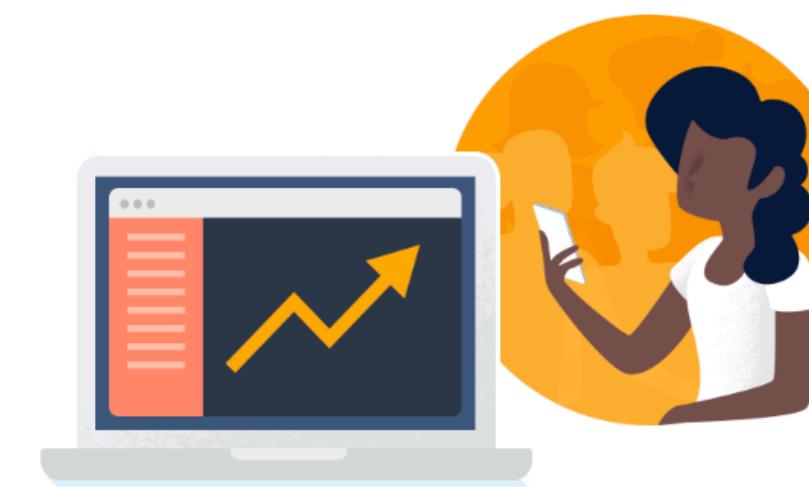
Test Lab

Test your app on devices hosted by Google



App Distribution BETA

Distribute pre-release versions of your app to your trusted testers



Grow your business



In-App Messaging BETA

Engage active app users with contextual messages



Google Analytics

Get free and unlimited app analytics



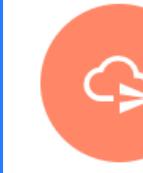
Predictions

Smart user segmentation based on predicted behavior



A/B Testing BETA

Optimize your app experience through experimentation



Cloud Messaging

Send targeted messages and notifications



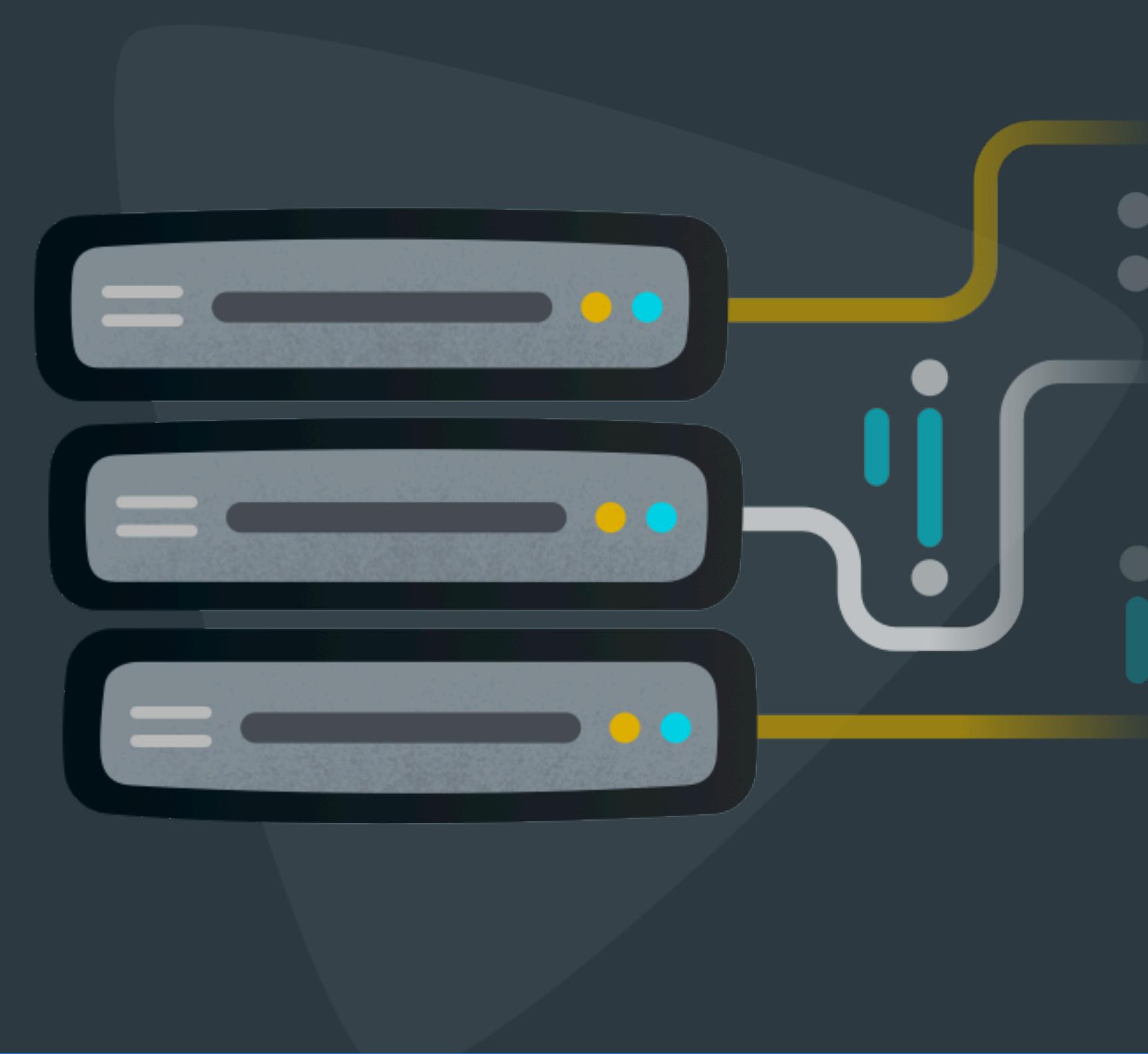
Remote Config

Modify your app without deploying a new version



Dynamic Links

Drive growth by using deep links with attribution



Realtime Database & REST API

Store and sync data in real time

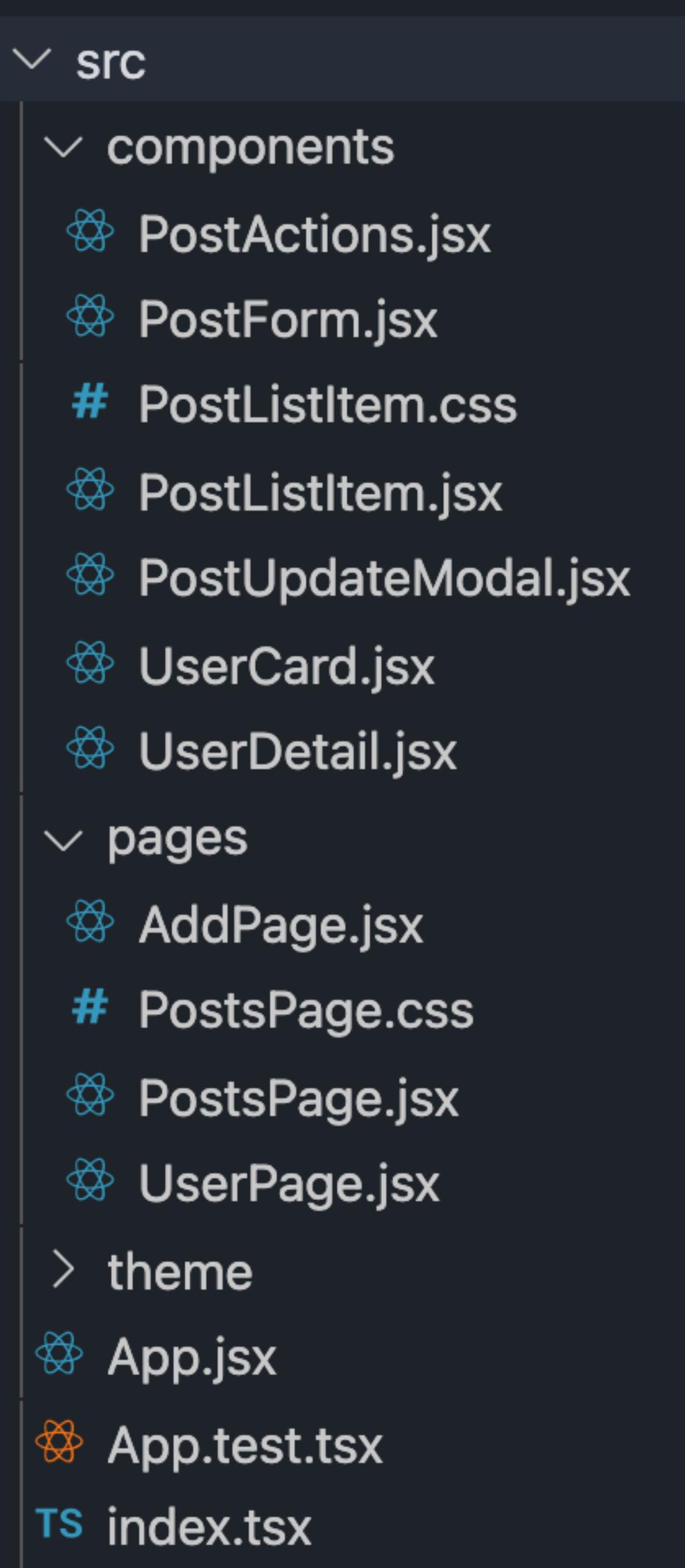
<https://firebase.google.com/products/realtime-database>

CRUD, CRUD, CRUD

... data, file storage, push notifications,
authentication, hosting, etc.

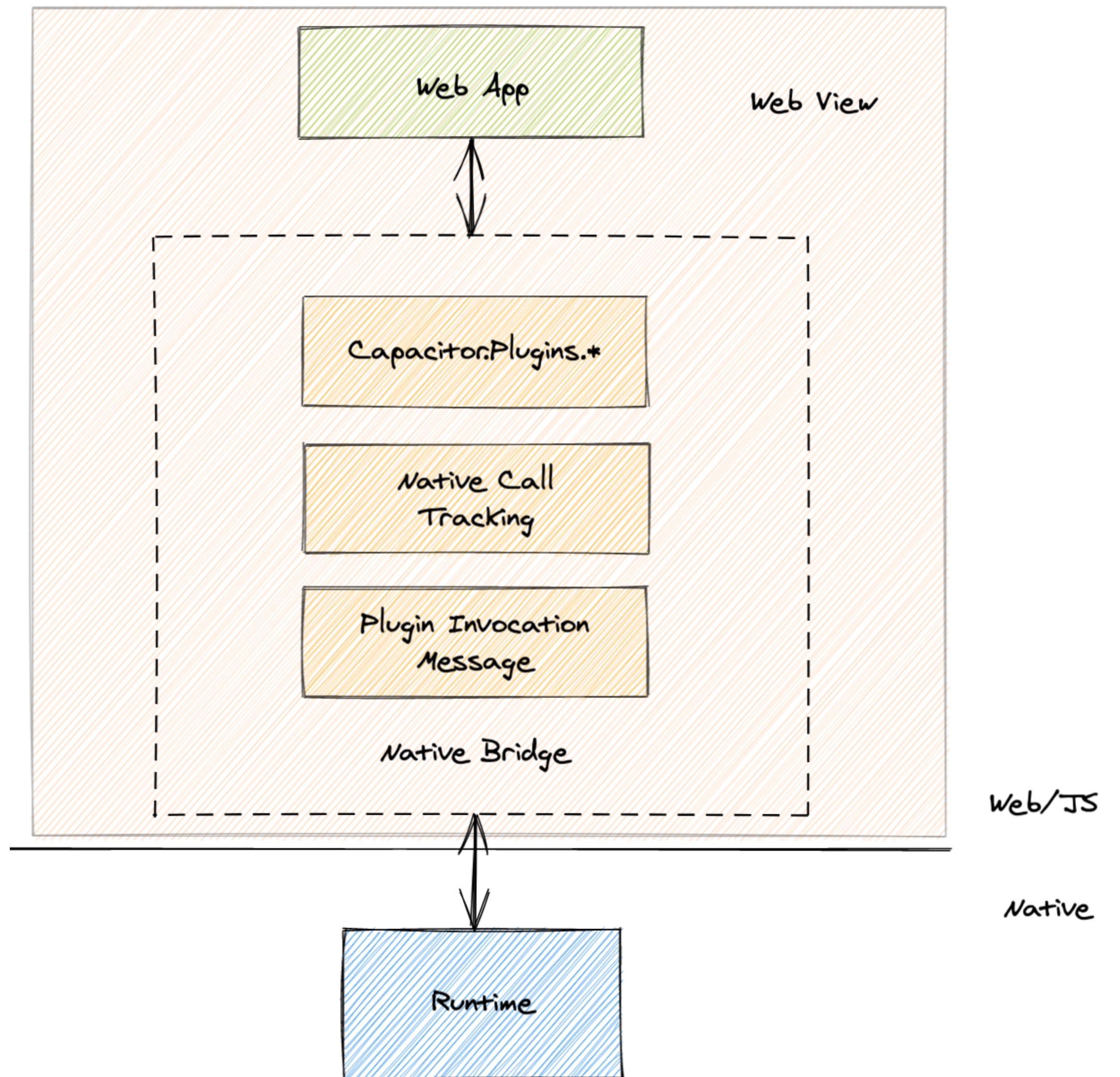
Thinking in React

In React, UI is a **function of props & states**. The UI is built with (function) **Components**.



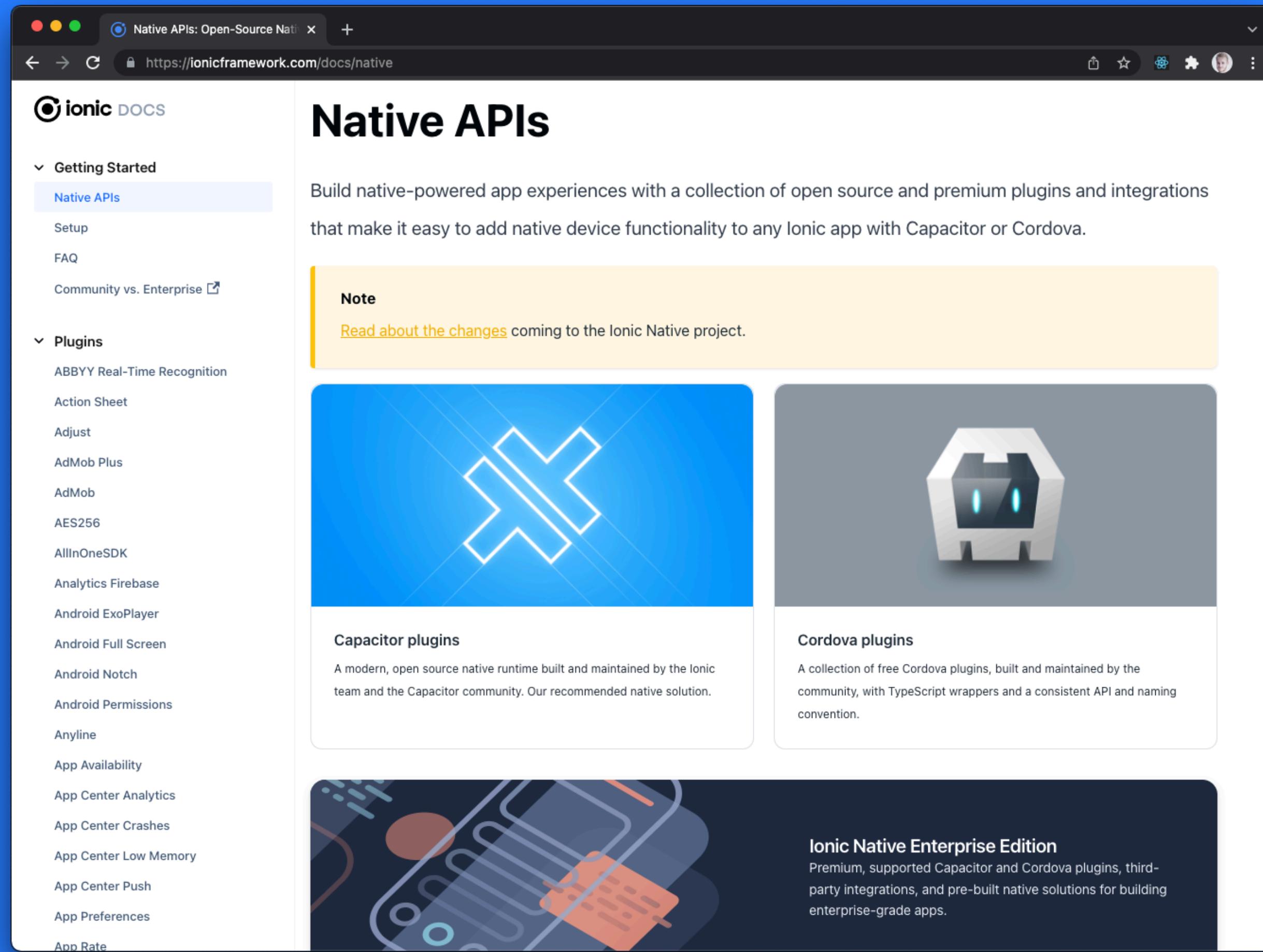
Native APIs

... provides access to native features such as camera, geolocation, and filesystem in your Web Native App.



Capacitor Plugins

Enables directly access to native devices features with Native APIs



<https://ionicframework.com/docs/native>

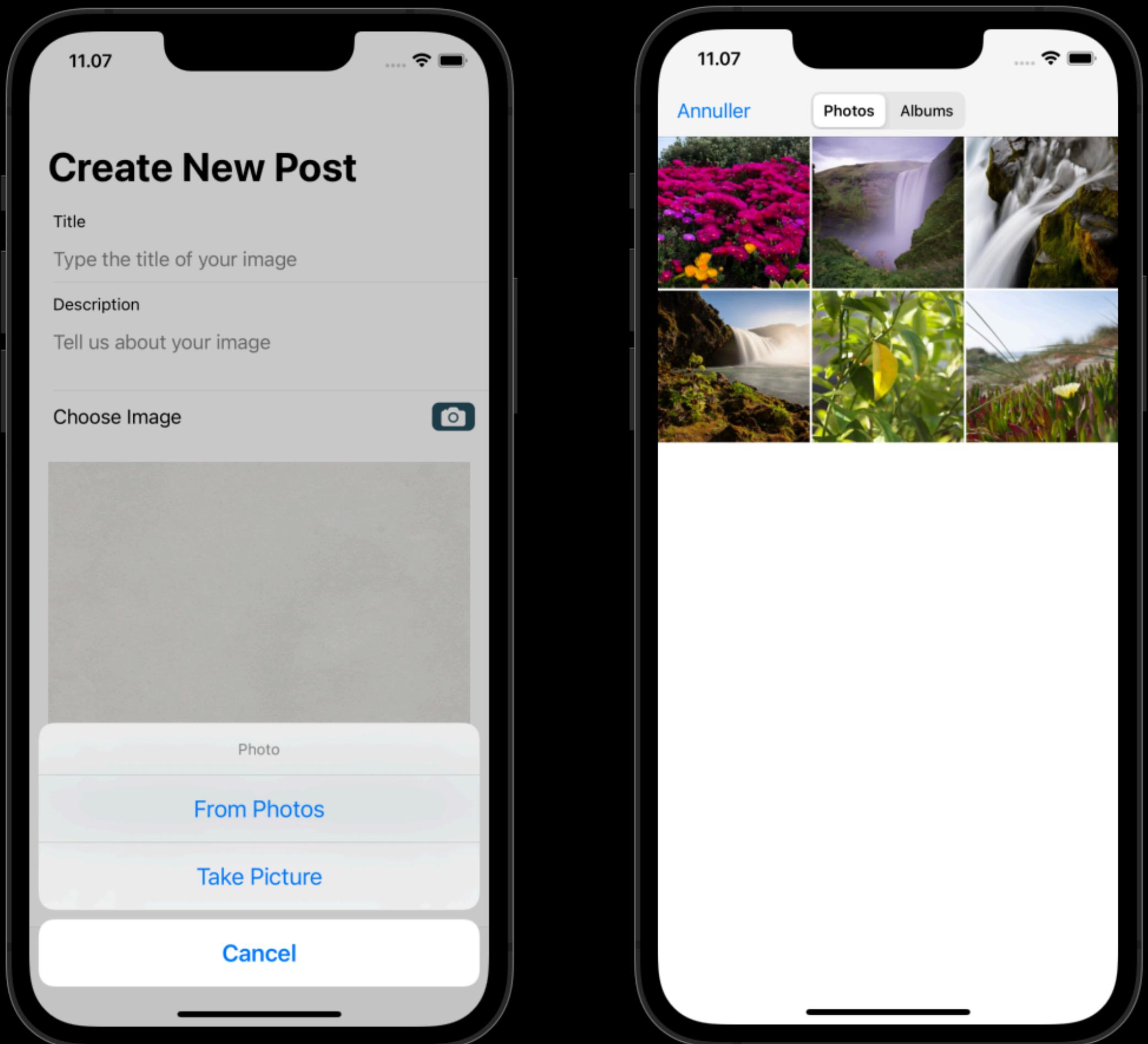
Capacitor

A cross-platform native runtime for Web Native Apps.

The Web View and the native app communicate through the use of Capacitor or Cordova plugins. Plugins provide native APIs such as camera, geolocation, and filesystem access to your web app.



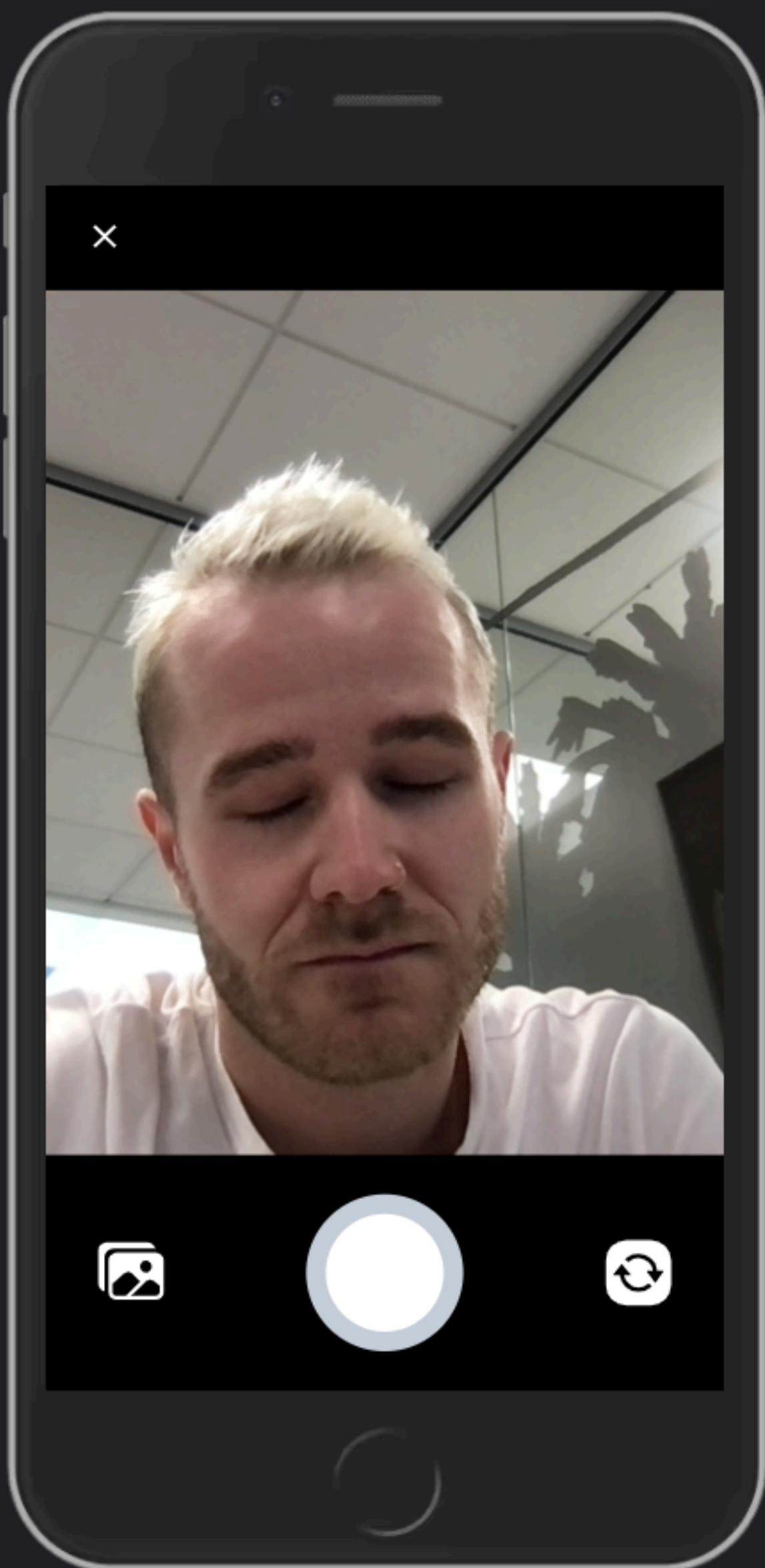
@capacitor/camera



```
async function takePicture() { ←  
  const imageOptions = {  
    quality: 80,  
    width: 500,  
    allowEditing: true,  
    resultType: CameraResultType.DataUrl  
  };  
  const image = await Camera.getPhoto(imageOptions);  
  const imageUrl = image.dataUrl;  
  setImage(imageUrl);  
  
  return (  
    <form onSubmit={submitEvent}>  
      <IonItem>  
        <IonLabel position="stacked">Title</IonLabel>  
        <IonInput value={title} placeholder="Type the title" />  
      </IonItem>  
      <IonItem>  
        <IonLabel position="stacked">Description</IonLabel>  
        <IonTextarea value={body} placeholder="Tell us about your image" />  
      </IonItem>  
      <IonItem onClick={takePicture} lines="none">  
        <IonLabel>Choose Image</IonLabel>
```

@ionic/pwa-elements

Web-based UI for PWAs



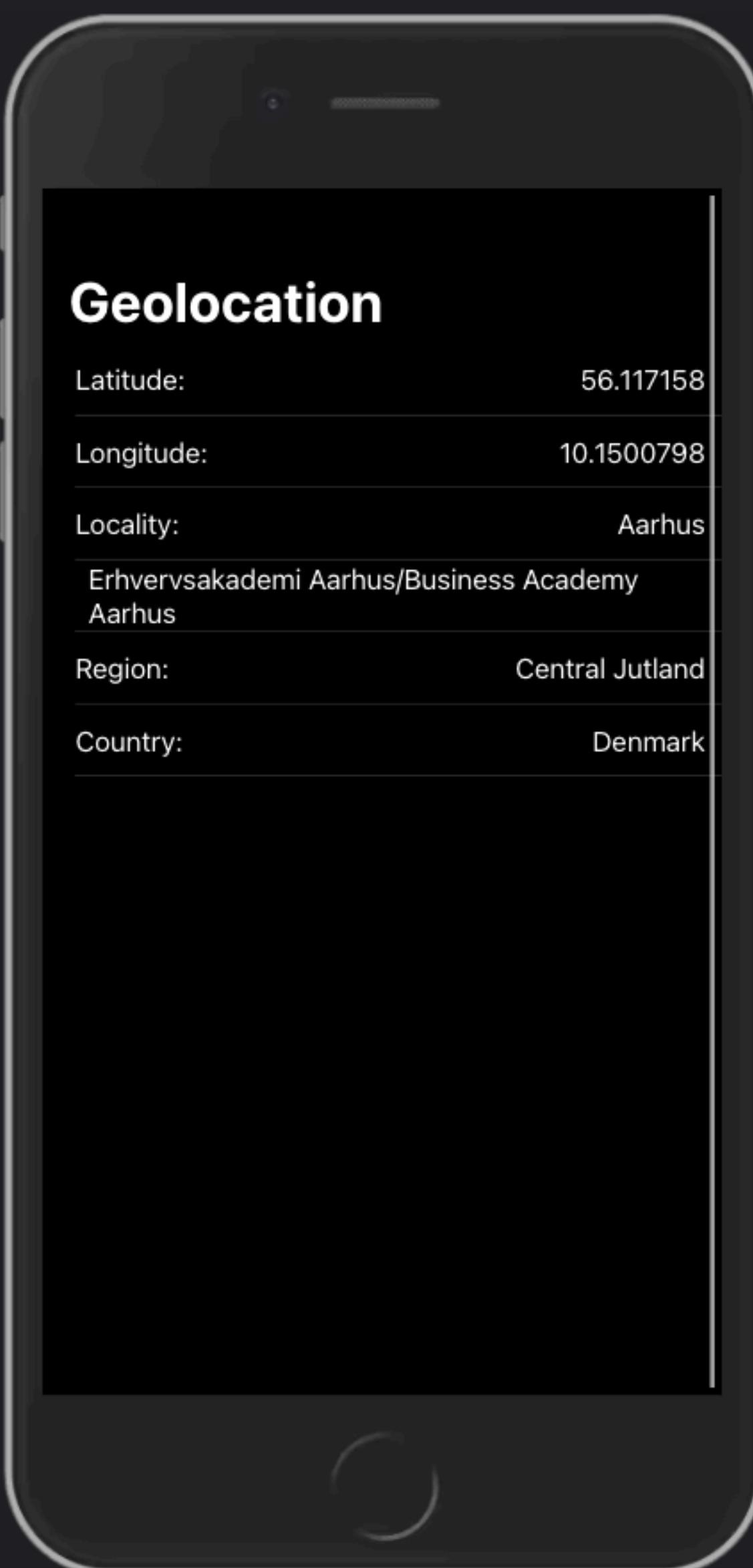
```
npm install @ionic/pwa-elements
```

```
import { defineCustomElements } from '@ionic/pwa-elements/loader';

ReactDOM.render(<App />, document.getElementById('root'));

// Call the element loader after the app has been rendered the first time
defineCustomElements(window);
```

@capacitor/geolocation



Track the current position of the device

```
import { IonContent, IonHeader, IonPage, IonTitle, IonToolbar, IonItem, IonLabel } from "@ionic/react"
import { Geolocation } from "@capacitor/geolocation";
import { useEffect, useState } from "react";

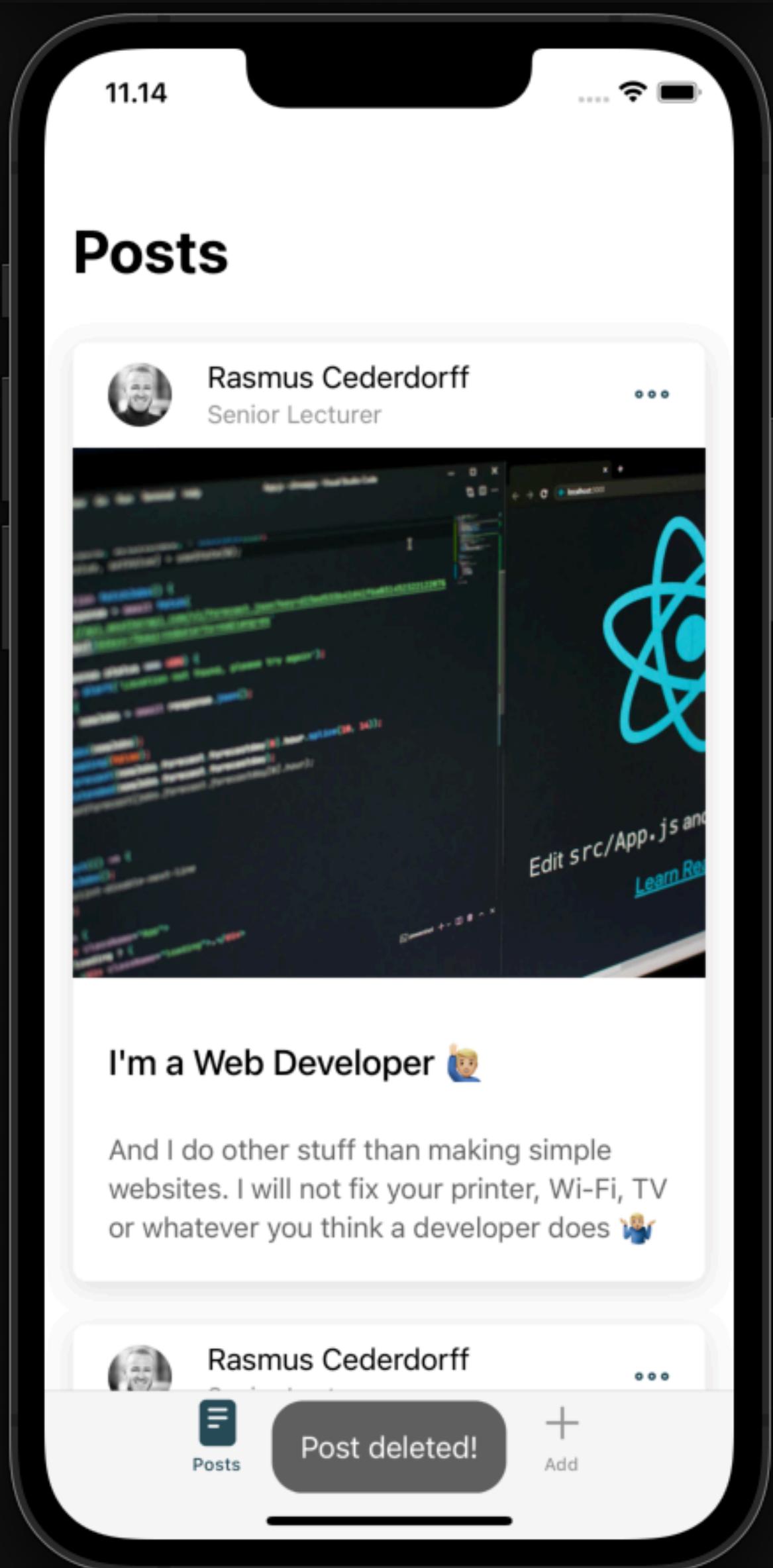
const Home = () => {
  const [lat, setLat] = useState("");
  const [long, setLong] = useState("");
  const [location, setLocation] = useState("");

  async function getCurrentPosition() {
    const coordinates = await Geolocation.getCurrentPosition();
    console.log("Current position:", coordinates);
    setLat(coordinates.coords.latitude);
    setLong(coordinates.coords.longitude);
    getLocation(coordinates.coords.latitude, coordinates.coords.longitude);
  }

  async function getLocation(latitude, longitude) {
    const key = "14de60fd94d4d8753fe8a277ba88667a";
    const res = await fetch(`http://api.positionstack.com/v1/reverse?access_key=${key}&lat=${latitude}&lon=${longitude}`);
    const result = await res.json();
    console.log(result);
    const loc = result.data[0];
    setLocation(loc);
  }
}
```

@capacitor/toast

Pop up notification



```
import { Toast } from '@capacitor/toast';

const showHelloToast = async () => {
  await Toast.show({
    text: 'Hello!',
  });
};
```

Icons & Splash Screens

<https://race.notion.site/Splash-Screen-and-Icons-be395d6b4b8b49e896797d1eb36e2fb2>

Next Thursday

The screenshot shows a Notion page with a yellow header bar featuring the Notion logo and several small icons related to app development. The main title of the page is "5. Firebase & App Dev". Below the title, there are three data cards: Date (03/03/2022), Teacher (RACE), and Course (MAD). The "Themes" section lists the following topics:

- Firebase
- Authentication & User Management
- Cloud Storage & Push Notifications
- Ionic Project
- Recap on chosen themes: <https://eaaa.padlet.org/race/mad>
- Next Steps