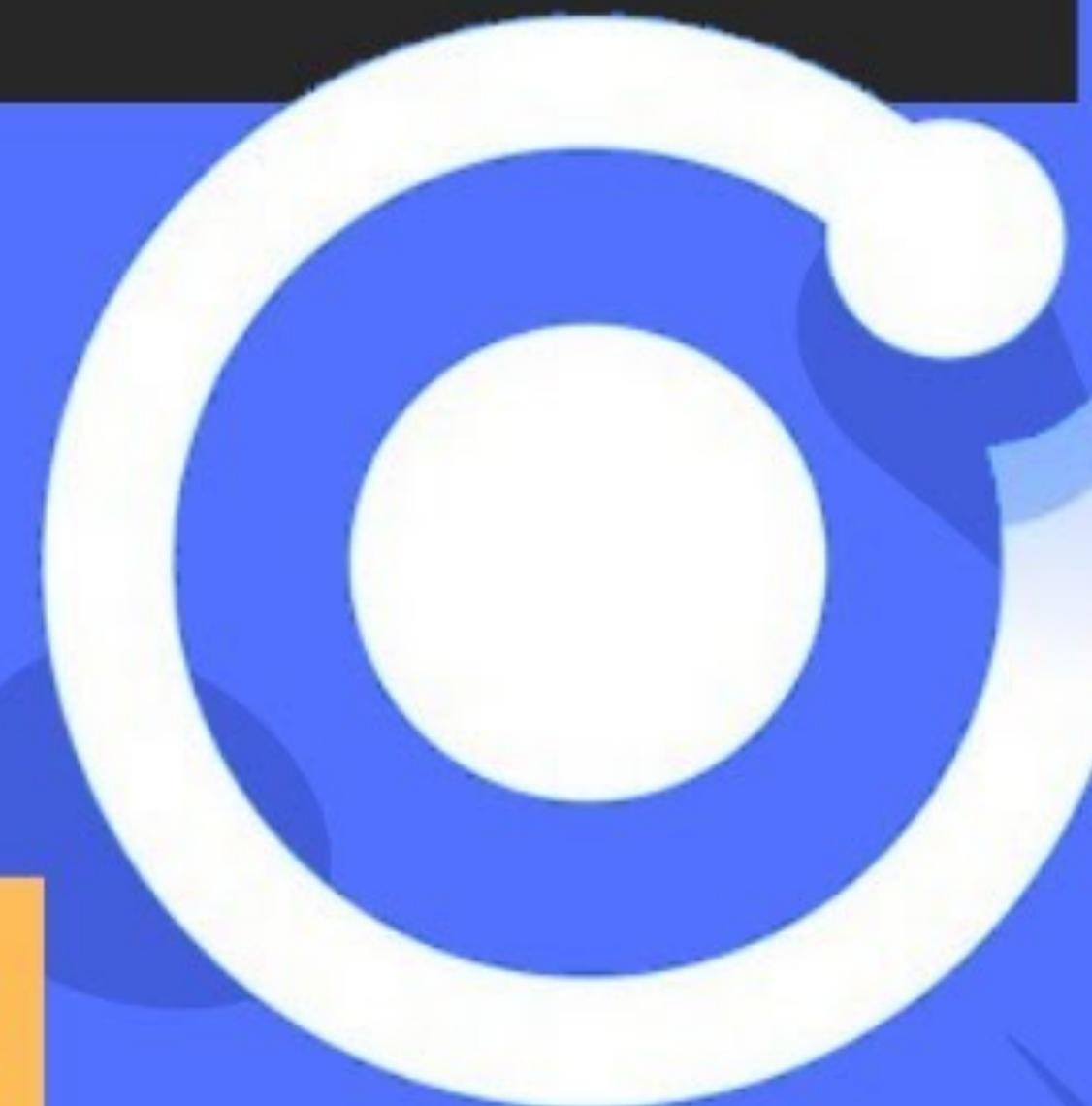


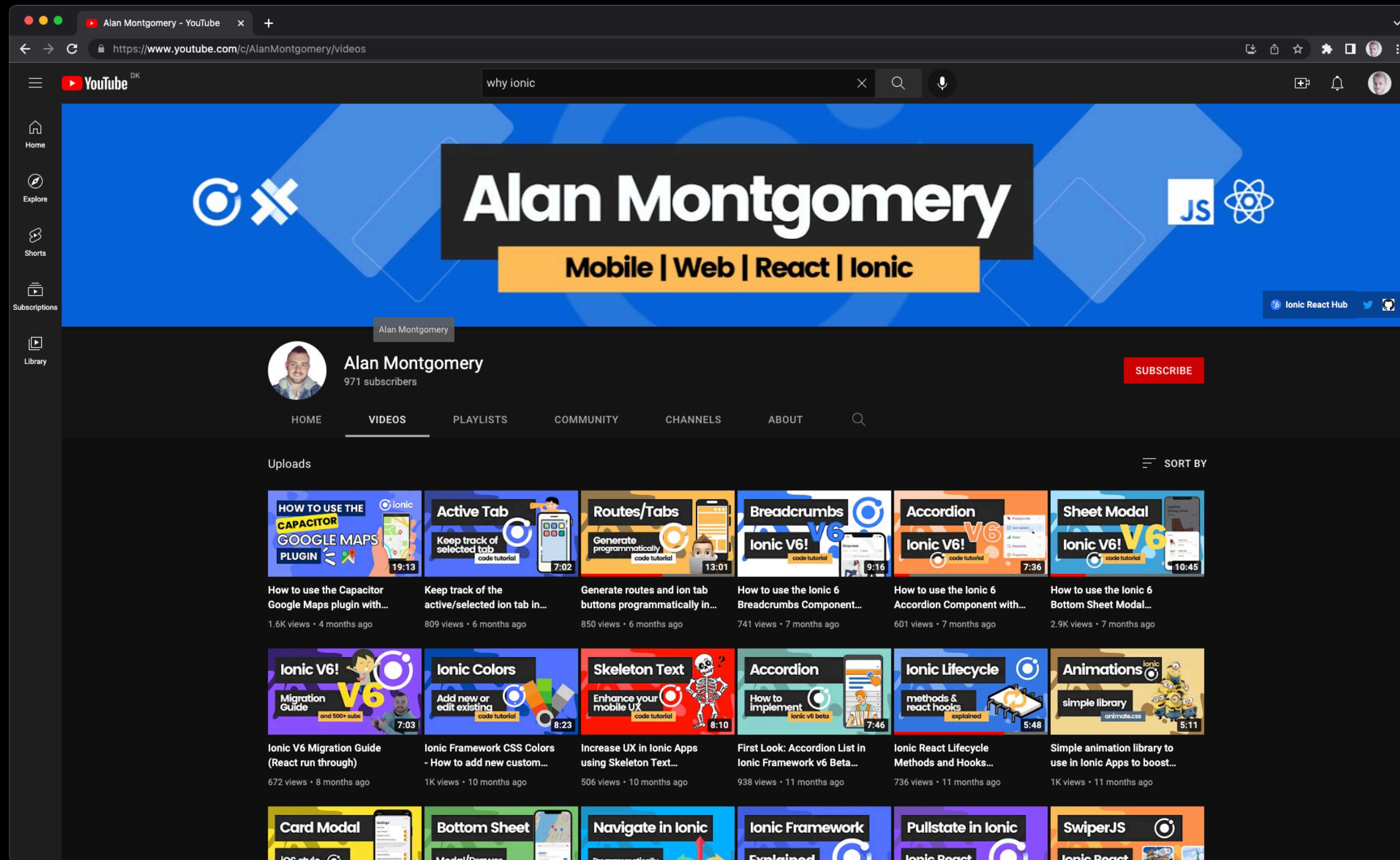
Ionic Framework

Explained

in 60 seconds



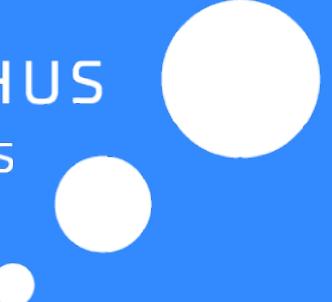
Native Device Features



<https://www.youtube.com/c/AlanMontgomery/videos>

Native Device Features

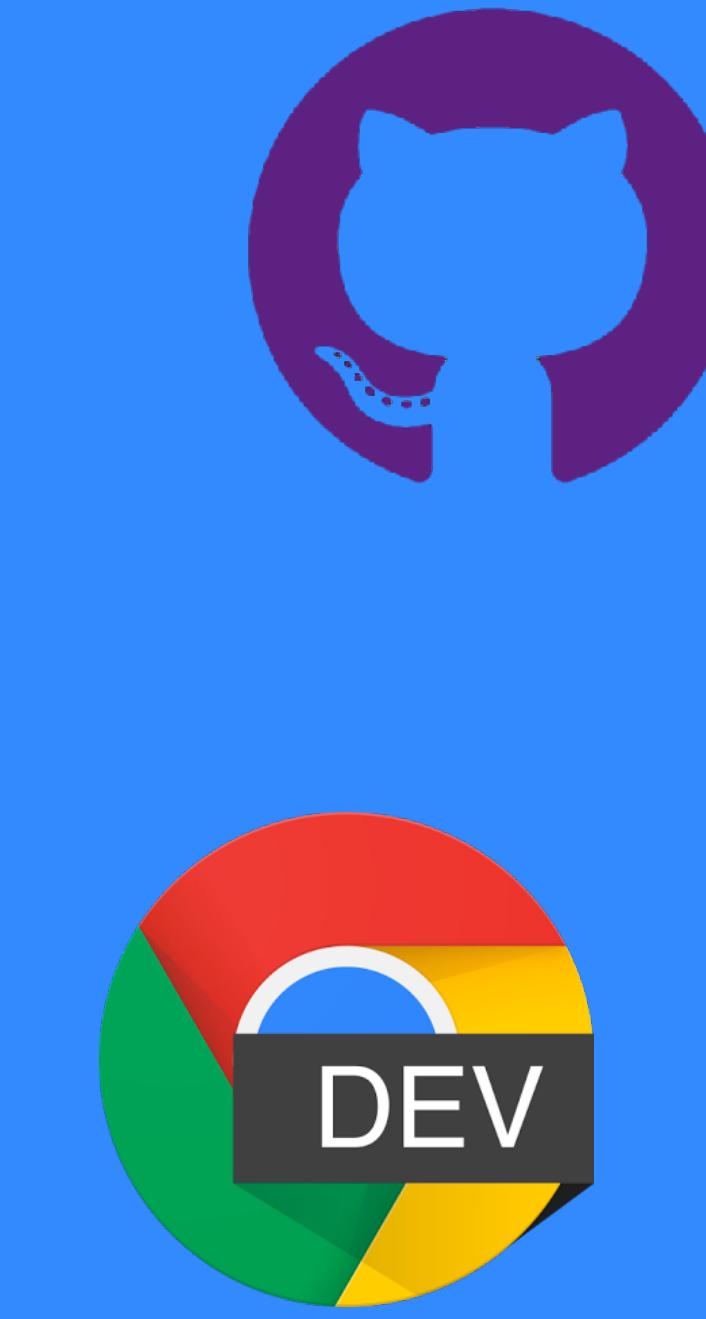
Build for platforms



Agenda

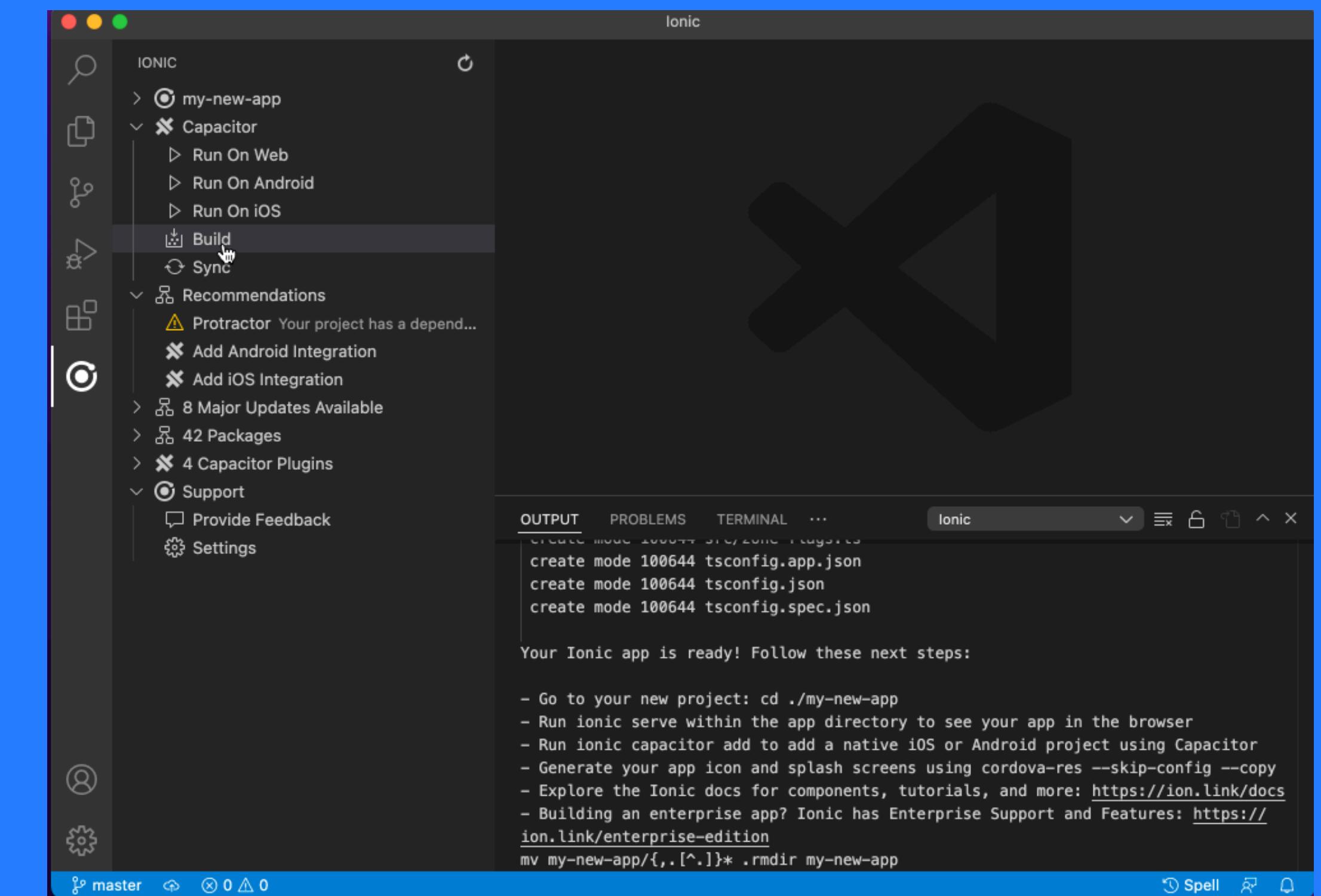
- Workflow & Dev Tips
- Ionic Post App
- Capacitor JS & Native API's
- Build for Platforms
- Ionic App Project

Workflow



Workflow & Tips

- Open in Integrated Terminal || cd into projects
- Run existing projects && npm install
- Ionic CLI: `ionic start` && `ionic serve`
- Keybindings
- Auto imports / fix imports
- Try Ionic VS Code Extensions



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>

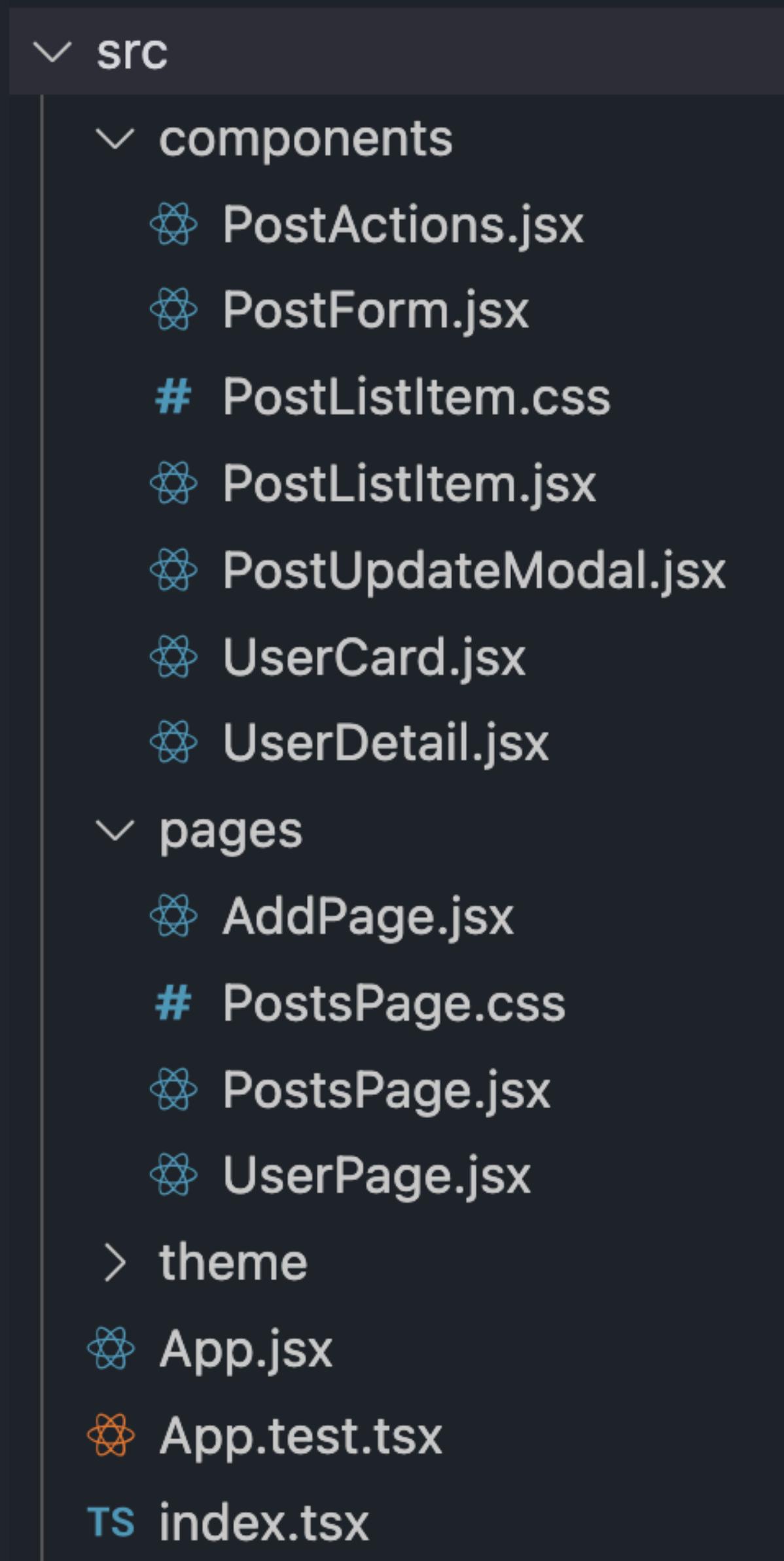
<https://ionicframework.com/docs/react/your-first-app/deploying-mobile>

Ionic Post App

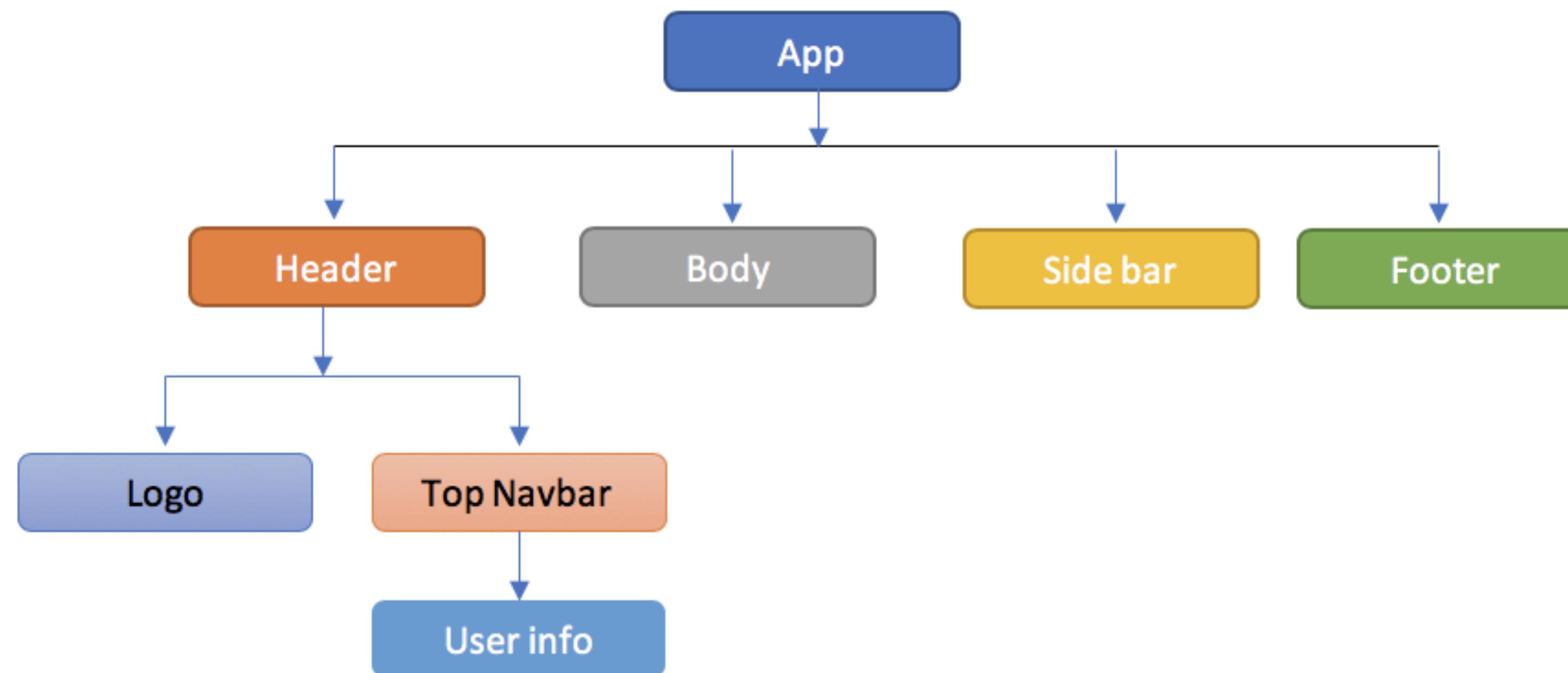
ionic-post-app*

Thinking in React

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



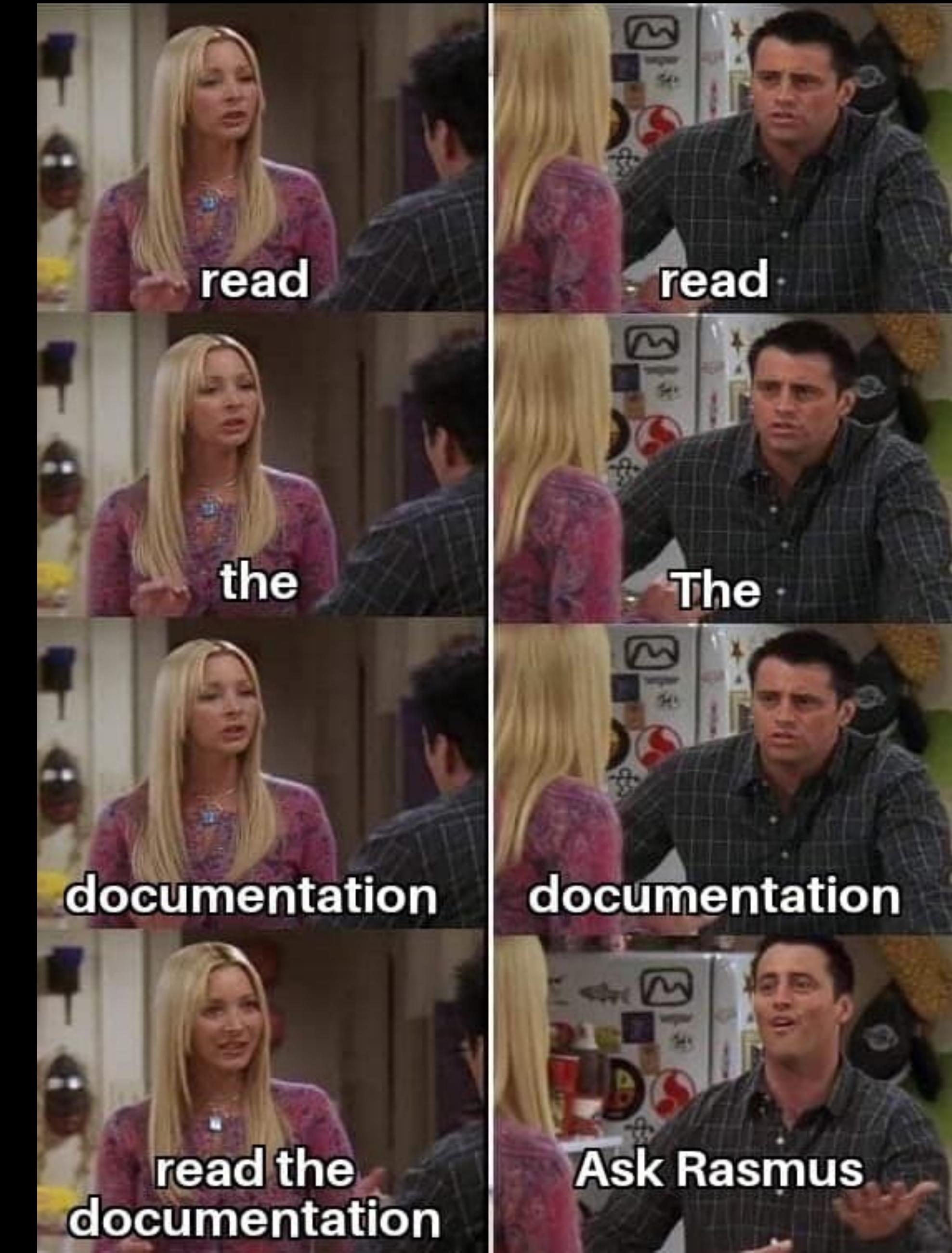
APP STRUCTURED IN COMPONENTS



UI Components

The screenshot shows a web browser window displaying the Ionic UI Components documentation at ionicframework.com/docs/components. The page has a dark header with the Ionic logo and navigation links for Guide, Components (which is the active tab), CLI, Native, and an Upgrade Guide. The main content area features a large title "UI Components" and a paragraph explaining that Ionic apps are built using high-level building blocks called Components. Below this, there's a grid of cards, each representing a different UI component:

- Action Sheet**: Shows a card with three horizontal bars. Description: Action Sheets display a set of options with the ability to confirm or cancel an action.
- Alert**: Shows a card with a list icon and a red badge with the number 2. Description: Alerts are a great way to offer the user the ability to choose a specific action or list of actions.
- Badge**: Shows a card with a red badge icon. Description: Badges are a small component that typically communicate a numerical value to the user.
- Button**: Shows a card with a blue button icon. Description: Buttons let your users take action. They're an essential way to interact with and navigate through an app.
- Card**: Shows a card with a card icon. Description: Cards are a great way to display an important piece of content, and can contain images, buttons, text, and more.
- Checkbox**: Shows a card with a checked checkbox icon. Description: Checkboxes can be used to let the user know they need to make a binary decision.
- Chip**: Shows a card with a chip icon. Description: Chips are a compact way to display data or actions.
- Content**: Shows a card with a smartphone icon. Description: Content is the quintessential way to interact with and navigate through an app.





What's...

- Ionic?
- React?
- Ionic React?
- Web Native?
- Capacitor JS?
- Firebase?

Open-Source UI Toolkit to Create

https://ionicframework.com/docs

ionic DOCS

Guide Components CLI Native Ionic v6.0.0 Upgrade Guide → v6 Search Community Support ☰

Getting Started Overview Upgrading to Ionic 6 Environment Setup CLI Installation Packages & CDN Ionic VS Code Extension Next Steps

Developing Starting Previewing Scaffolding Developing for iOS Developing for Android Development Tips Hardware Back Button Keyboard

Layout Structure Responsive Grid Global Stylesheets CSS Utilities

Theming Basics Platform Styles CSS Variables CSS Shadow Parts Colors Themes Dark Mode

Introduction to Ionic

iOS Android

Ionic is an open source UI toolkit for building performant, high-quality mobile apps using web technologies — HTML, CSS, and JavaScript — with integrations for popular frameworks like [Angular](#), [React](#), and [Vue](#).

Get started building by [installing Ionic](#) or following our [First App Tutorial](#) to learn the main concepts.

Installation Guide
Step-by-step guides to setting up your system and installing the framework.

UI Components
Dive into Ionic beautifully designed UI component library.

Native Functionality
Integrate native device plugins, like Bluetooth, Maps, HealthKit, and more.

Theming
Learn to easily customize and modify your Ionic app's visual design to fit your brand.

Overview

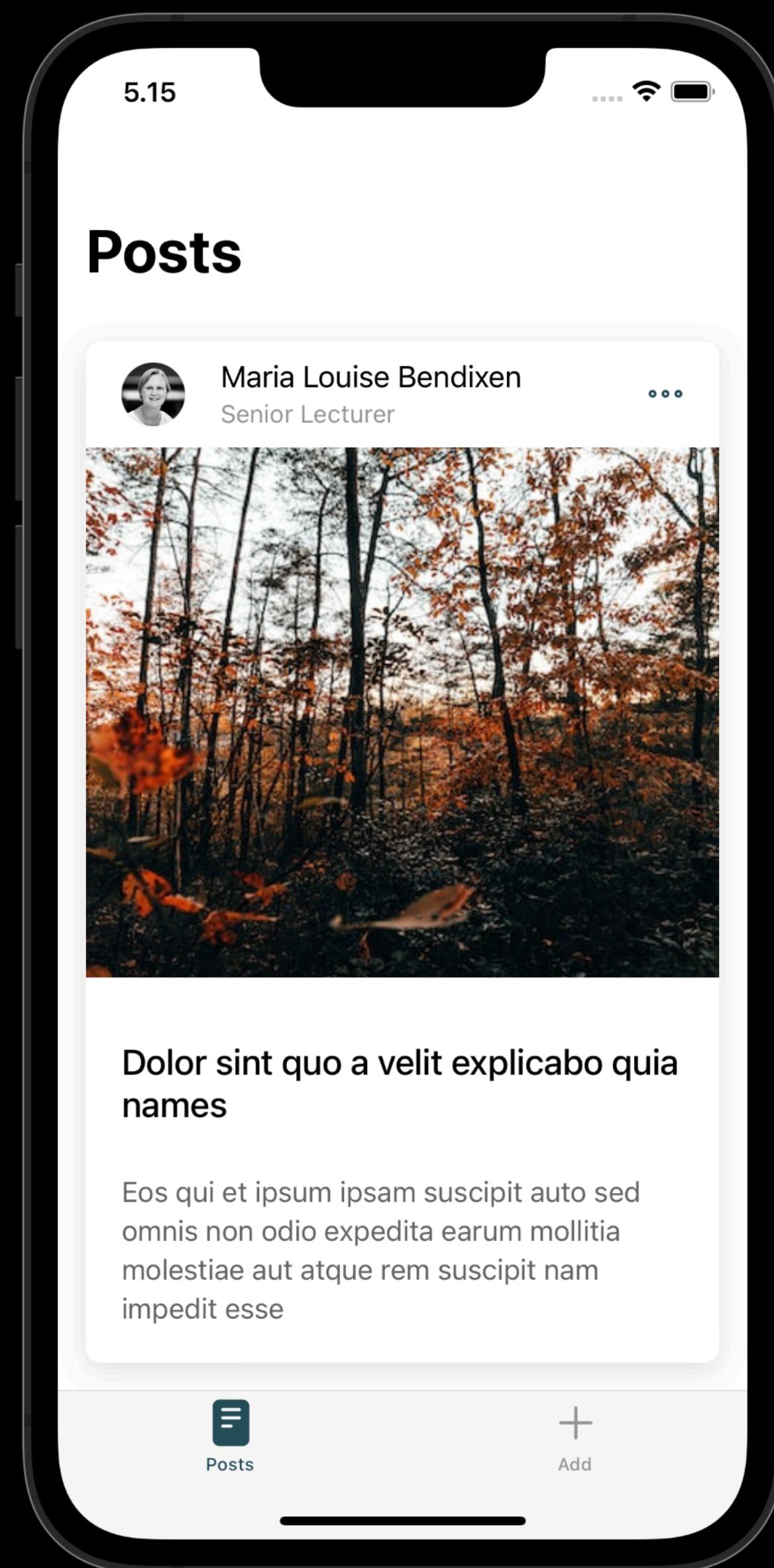
Ionic focuses on the frontend UX and UI interaction of an app — UI controls, interactions, gestures, animations. It's easy to learn, and integrates with other libraries or frameworks, such as [Angular](#), [React](#), or [Vue](#). Alternatively, it can be used standalone without any frontend framework using a simple `ionic build`. If you'd like to learn more about Ionic before diving

Ionic Framework

Dark Mode

- Accordion
- Action Sheet
- Alert
- Avatar
- Badge
- Breadcrumbs
- Button
- Card
- Checkbox
- Chip
- Content

[View Source](#)

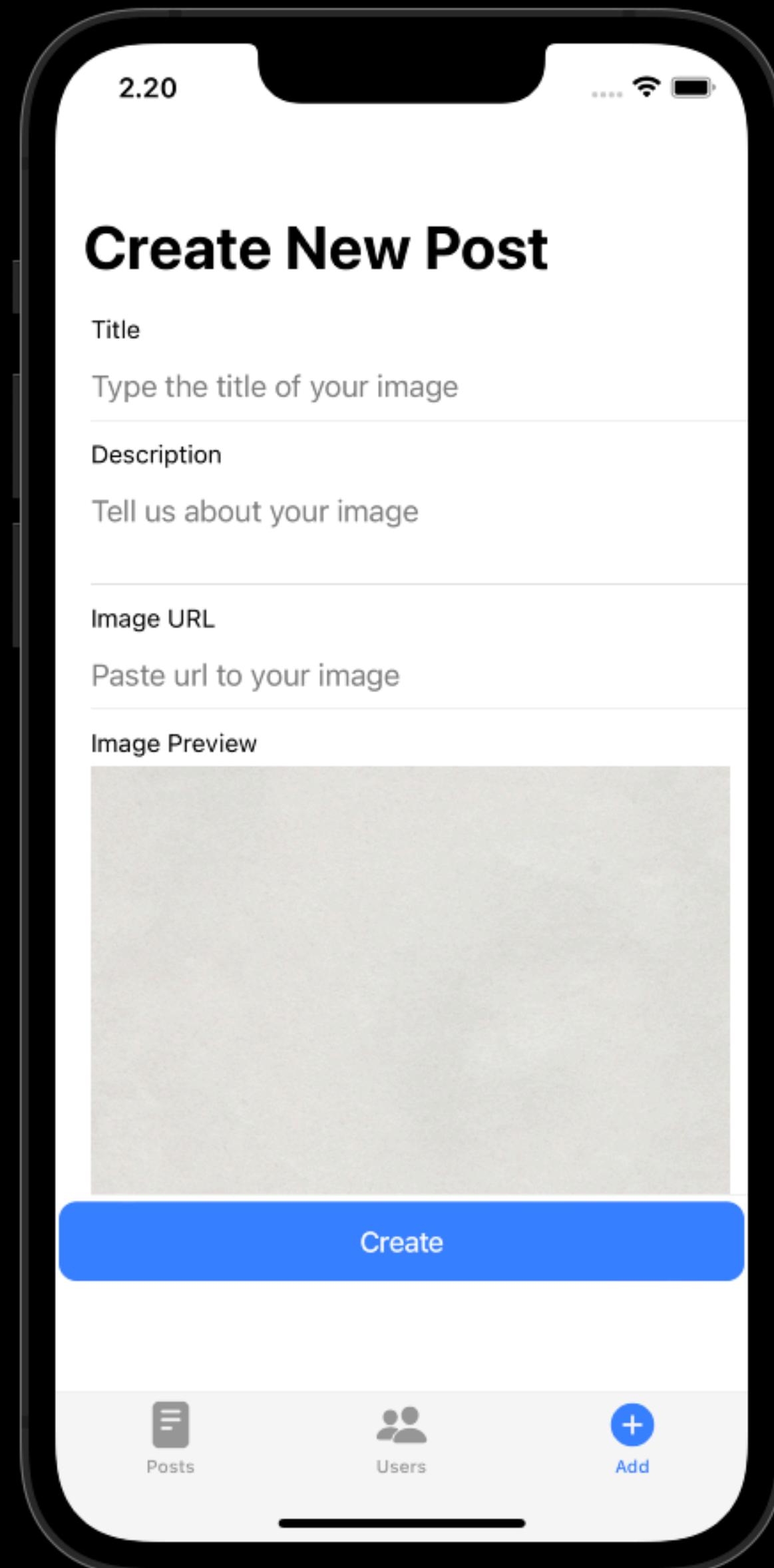


Ionic Post App w/ Firebase REST

Step 1 - 5

States & Forms

useState & onChange (onIonChange)



```
export default function AddPage() {  
  const [title, setTitle] = useState("");  
  const [body, setBody] = useState("");  
  const [url, setUrl] = useState("");  
  
  // ...  
  
  <form onSubmit={handleSubmit}>  
    <IonItem>  
      <IonLabel position="stacked">Title</IonLabel>  
      <IonInput value={title} placeholder="Type the title of your image"  
        onIonChange={e => setTitle(e.target.value)} />  
    </IonItem>  
    <IonItem>  
      <IonLabel position="stacked">Description</IonLabel>  
      <IonTextarea value={body} placeholder="Tell us about your image"  
        onIonChange={e => setBody(e.target.value)}></IonTextarea>  
    </IonItem>  
    <IonItem>  
      <IonLabel position="stacked">Image URL</IonLabel>  
      <IonInput value={url} type="url" placeholder="Paste url to your image"  
        onIonChange={e => setUrl(e.target.value)} />  
    </IonItem>  
    <IonItem>  
      <IonLabel position="stacked">Image Preview</IonLabel>  
      <IonImg src={url === "" ? fallbackUrl : url} />  
    </IonItem>  
    <IonButton type="submit" expand="block">
```

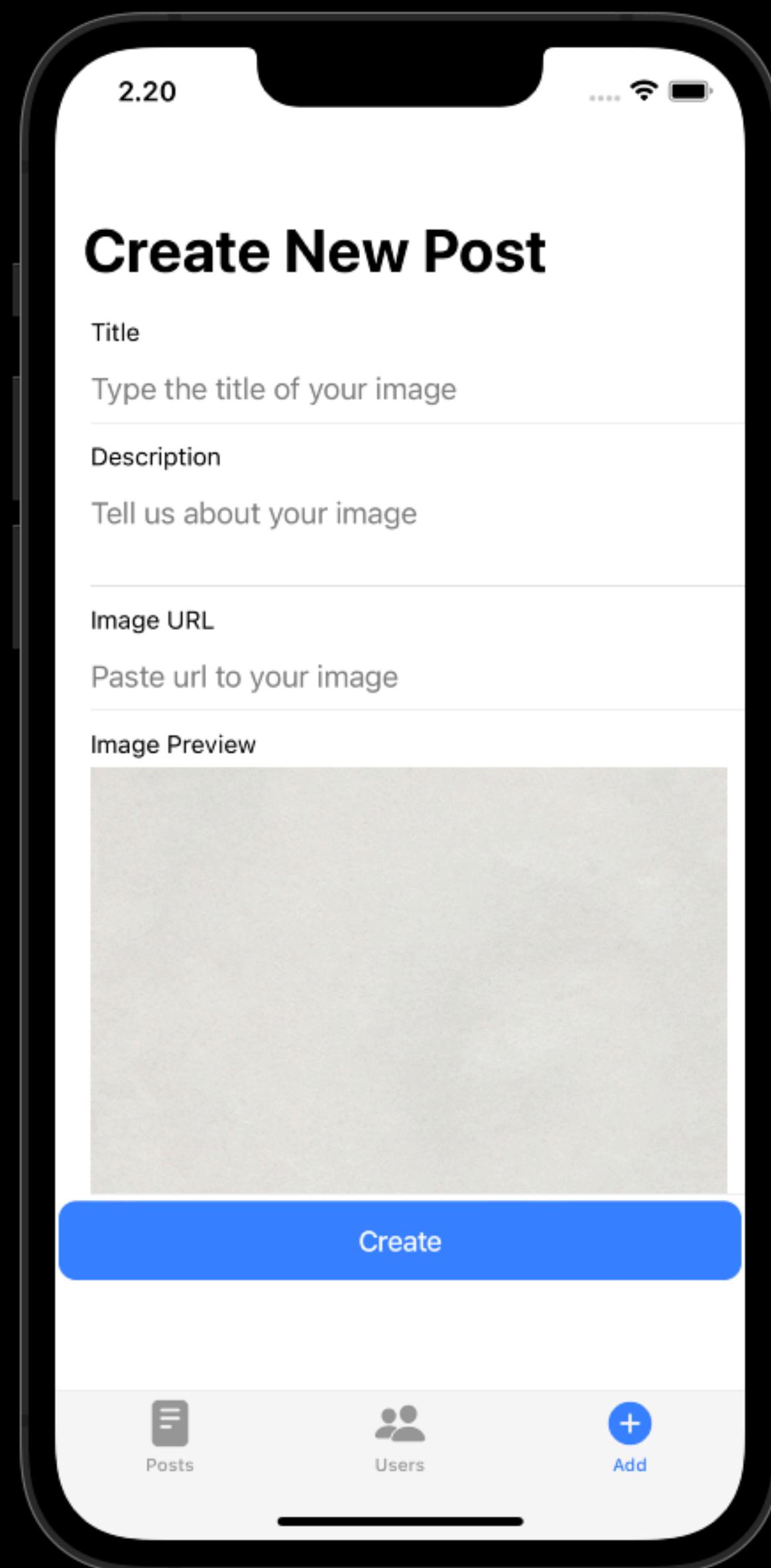
Form Object

```
export default function FormStateObject() {
  const [formData, setFormData] = React.useState({ firstName: "", lastName: "", mail: "" });

  function handleChange(event) {
    const name = event.target.name;
    const value = event.target.value;
    setFormData(prevFormData => {
      return {
        ...prevFormData,
        [name]: value
      };
    });
  }

  return (
    <form>
      <input type="text" placeholder="First Name" onChange={handleChange} name="firstName" />
      <input type="text" placeholder="Last Name" onChange={handleChange} name="lastName" />
      <input type="email" placeholder="Mail" onChange={handleChange} name="mail" />

      <p>Form data object: {JSON.stringify(formData)}</p>
    </form>
  );
}
```



```
export default function AddPage() {
  const [title, setTitle] = useState("");
  const [body, setBody] = useState("");
  const [url, setUrl] = useState("");
  const fallbackUrl = "https://media.istockphoto.com/photos/white-paper-texture-ba"

  function handleSubmit(event) { ←
    event.preventDefault();

    const newPost = {
      uid: 4,
      title: title,
      body: body,
      image: url
    };
    postService.createPost(newPost);

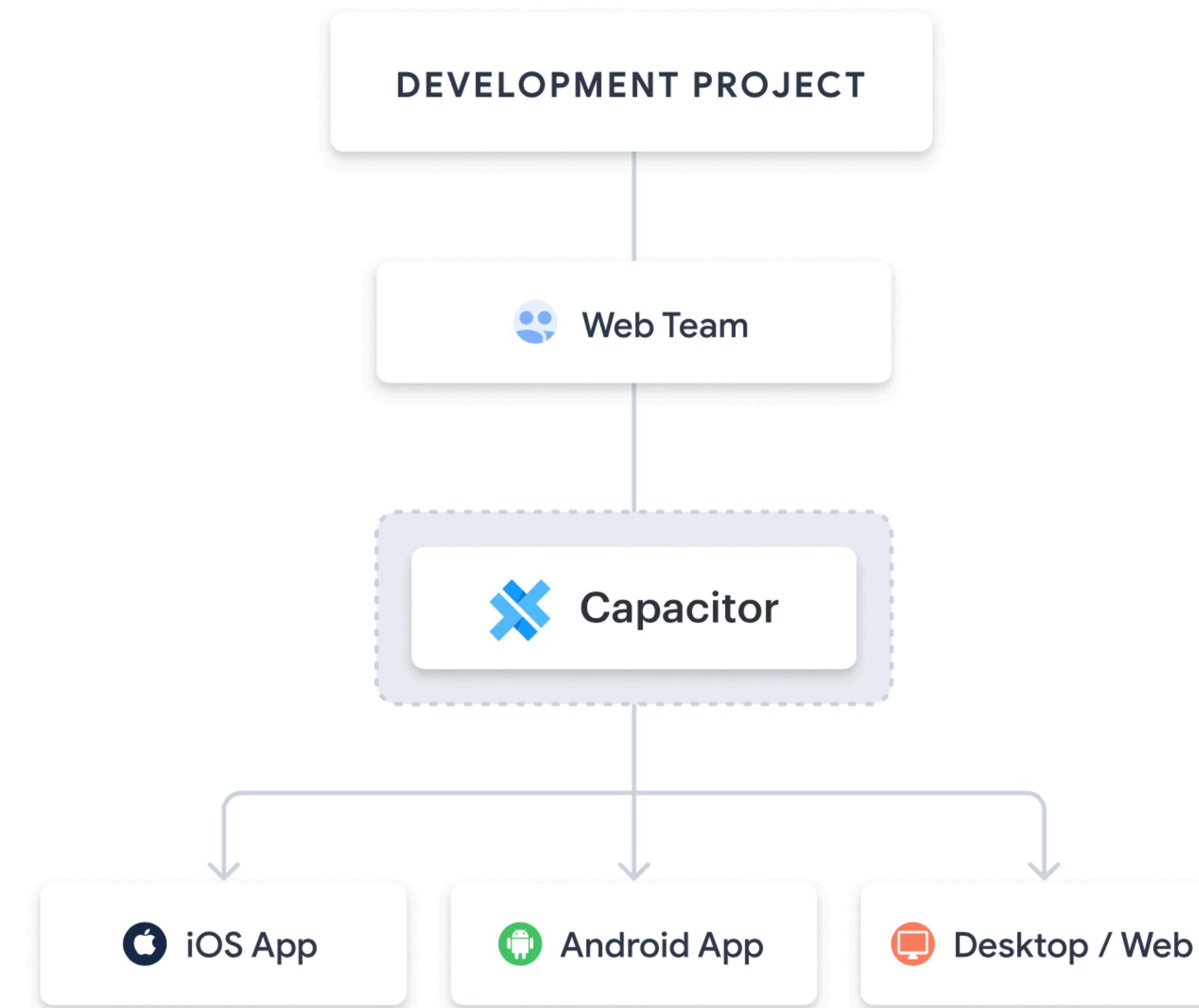
  }

  return (
    <IonPage>
      <IonHeader>...
      </IonHeader>
      <IonContent fullscreen>
        <IonHeader collapse="condense">...
        </IonHeader>
        <form onSubmit={handleSubmit}>
          <IonItem>
            <IonLabel position="stacked">Title</IonLabel>
```

Capacitor JS & Native API's

[Building Cross-platform Apps with Capacitor](#)

Capacitor JS Project Flow



Web Native

“Web Native is the idea that teams should build modern Web Apps and combine them with tools like Capacitor that unlock powerful Native APIs to the app for the platform its running on.”

“Web Native is the full capability and access to developers of the Web Platform, with the full functionality and performance benefits of traditional native apps.”

“Web Native is “hybrid” done right, and it's the future of mobile app development.”



Back in the 2010s (Cordova)

Feature	Native	Web-only	Hybrid
Device Access	Full	Limited	Limited
Performance	High	Medium to High	Low
Development Language	Platform Specific	HTML, CSS, Javascript	HTML, CSS, Javascript
Cross-Platform Support	No	Yes	Yes
User Experience	High	Medium to High	Low
Code Reuse	No	Yes	Yes



Runs web apps
across multiple
platforms

Hybrid Apps

Today (Capacitor)

Feature	Native	Web-only	Hybrid
Device Access	Full	Limited	Full (with plugins)
Performance	High	Medium to High	Medium to High
Development Language	Platform Specific	HTML, CSS, Javascript	HTML, CSS, Javascript
Cross-Platform Support	No	Yes	Yes
User Experience	High	Medium to High	Medium to High
Code Reuse	No	Yes	Yes



Runs modern web apps
natively on iOS,
Android, Electron and
Web multiple platforms.

Web Native Apps & PWA

Capacitor

A cross-platform native runtime for web apps.



Capacitor is a cross-platform native runtime that makes it easy to build modern web apps that run natively on iOS, Android, and the Web.

Representing the next evolution of Hybrid apps, Capacitor creates Web Native apps, providing a modern native container approach for teams who want to build web-first without sacrificing full access to native SDKs when they need it.

<https://capacitorjs.com/>

Capacitor

A cross-platform native runtime for web apps.

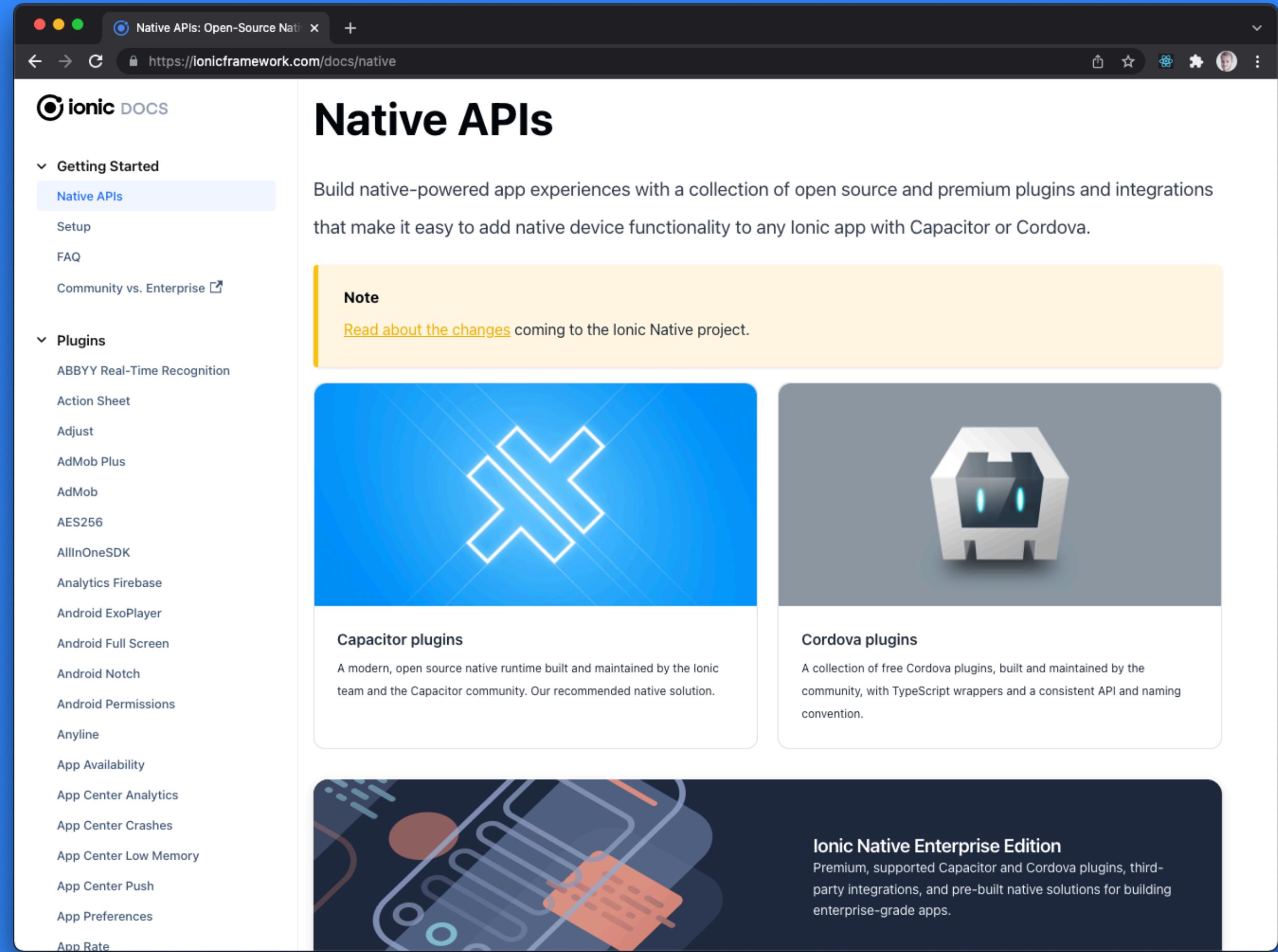
The Web View and the native app communicate through the use of Capacitor or Cordova plugins. Plugins provide native APIs such as camera, geolocation, and filesystem access to your web app.



<https://capacitorjs.com/>

Capacitor Plugins

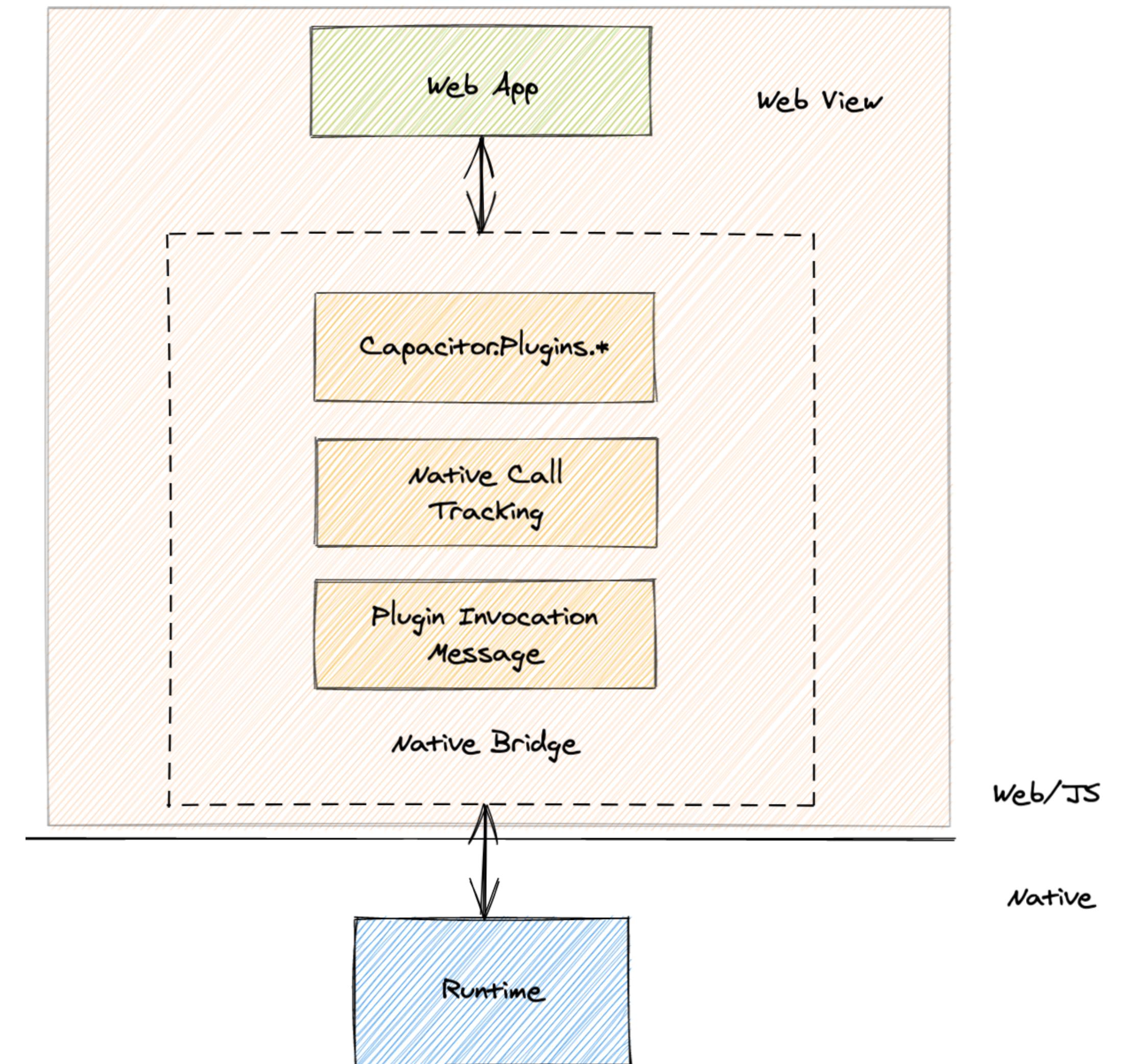
Enables directly access to native devices features with Native APIs



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

Native APIs

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



Build for Platforms

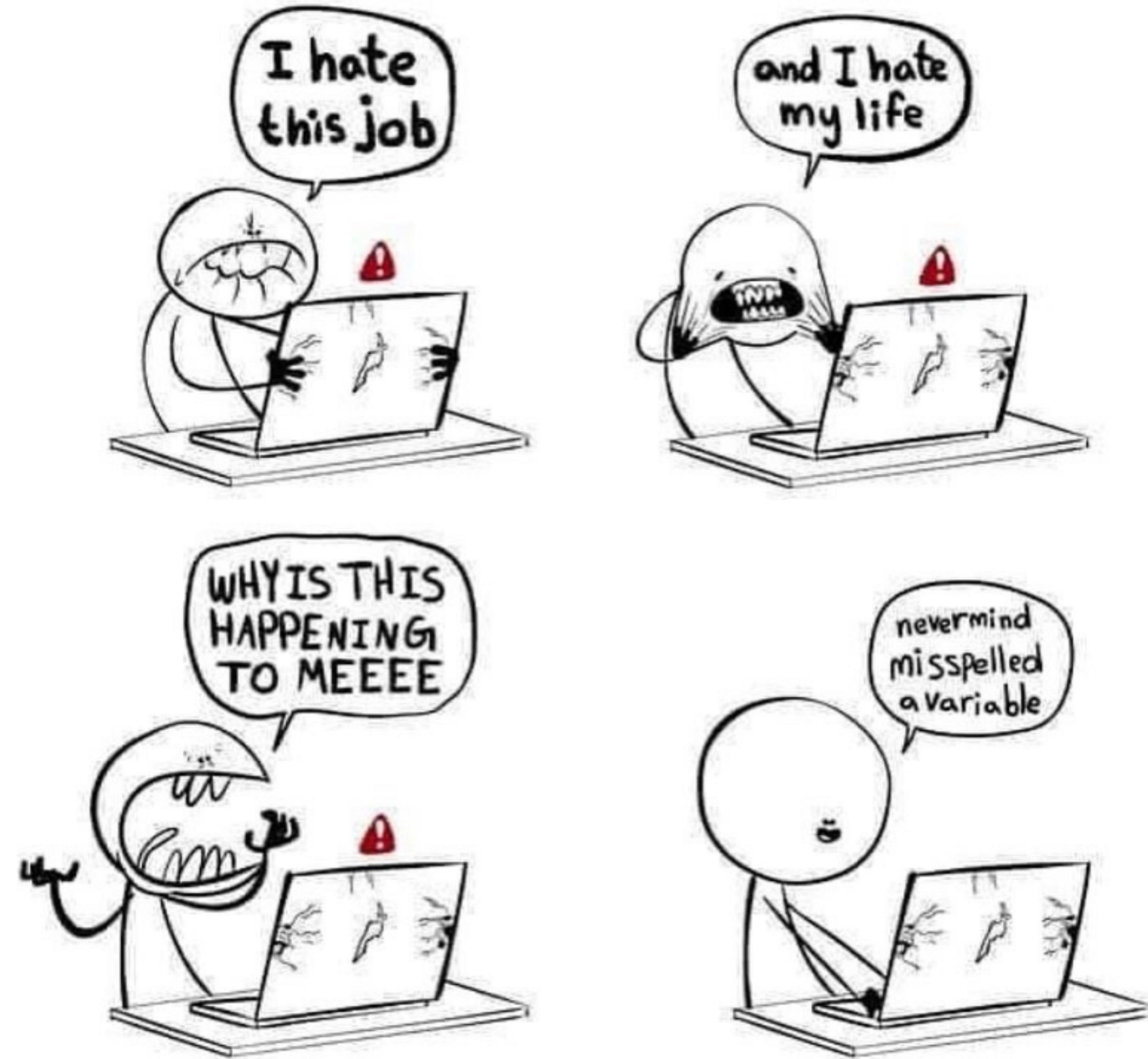
<https://ionicframework.com/docs/react/your-first-app/deploying-mobile>

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)

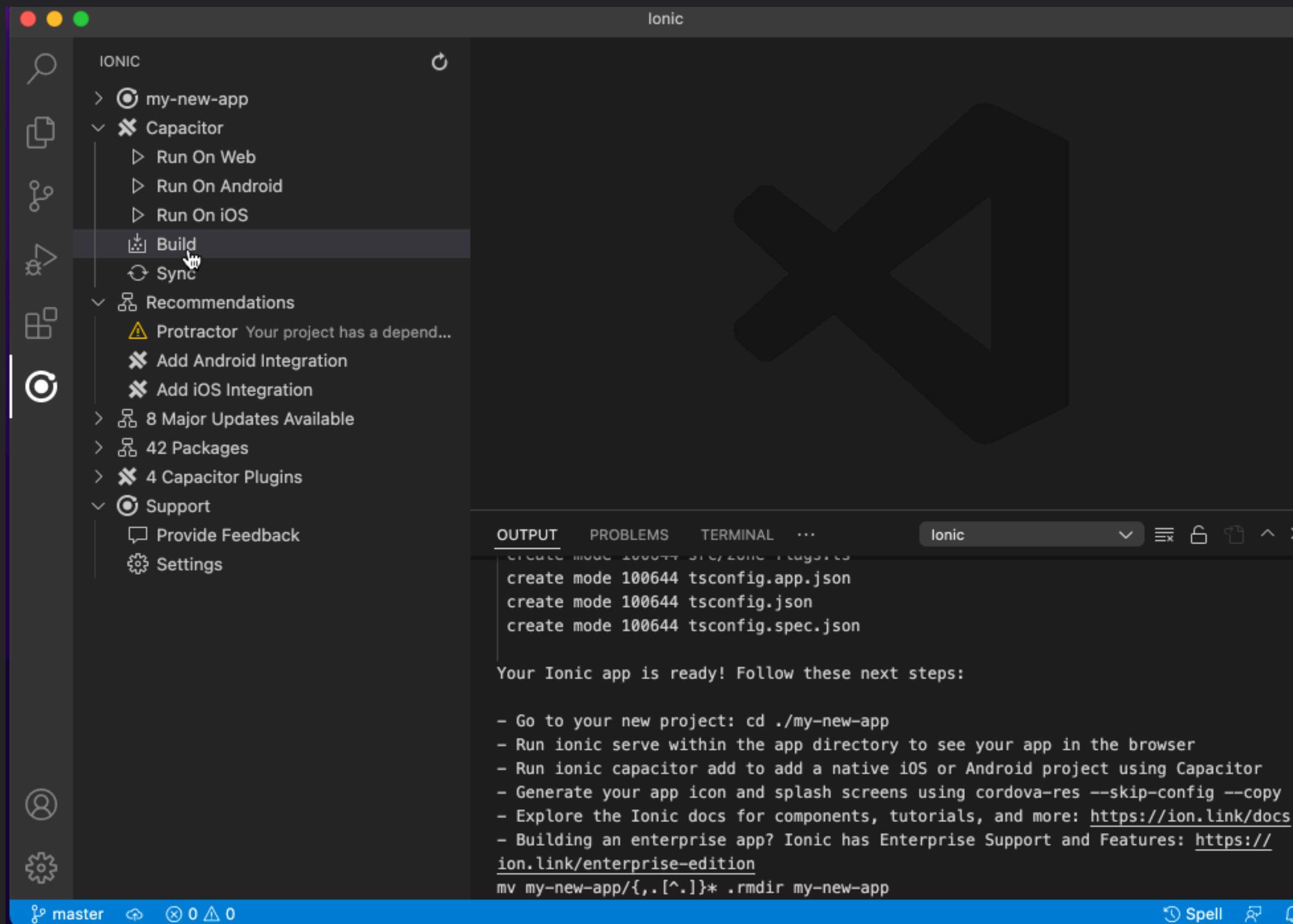
Deploying to iOS and Android



Well, ~~programming~~ configuration can be hard



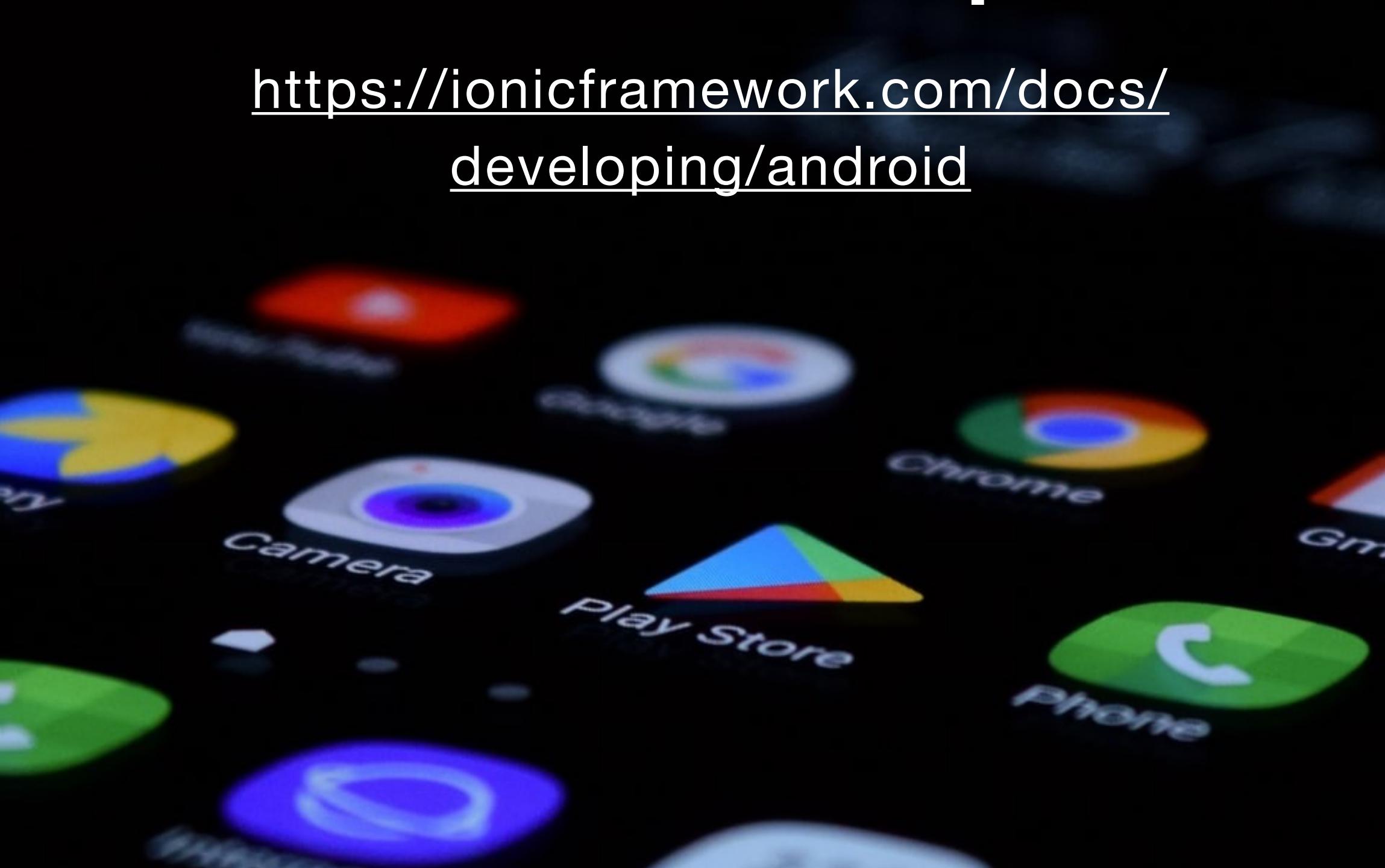
Ionic VS Code Extension



<https://capacitorjs.com/docs/getting-started/vscode-extension>

Android Development

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



You need Android Studio to run on Android emulators & devices.

Android can be developed on Windows, macOS & Linux.

Deploy to Android

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



Follow instructions for Capacitor only and use your existing Ionic Post App.

1. Install Android Studio and Android SDK.
2. Configure Command Line Tools (some environment variables must be set).
3. Create a Virtual device and/or set up Android Device.
4. Skip “Cordova Setup” and go to “Project Setup”.
5. Run with Capacitor (and skip “Running with Cordova”).
6. Try out debugging using Chrome DevTools.
7. Move on with Ionic Post App and use live reload to develop and test your app.

Dev tips

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

You need a Mac 🤦

Two main workflows:

1. Running with Xcode
2. Running with Ionic CLI /
use Ionic VS Code
Extension

iOS Development

[https://ionicframework.com/docs/
developing/ios](https://ionicframework.com/docs/developing/ios)

Follow instructions for Capacitor only and use your existing Ionic Post App.

1. Download Xcode from AppStore. Make sure you have the latest version.

2. Make sure the command-line tools are selected for use:

```
$ xcode-select --install
```

3. Add iOS with

```
$ ionic capacitor add ios
```

4. Open in Xcode and run with Xcode

5. Run with CLI & Live-reload with Capacitor

6. Debug iOS with Safari

7. Move on with Ionic Post App and use live reload to develop and test your app.

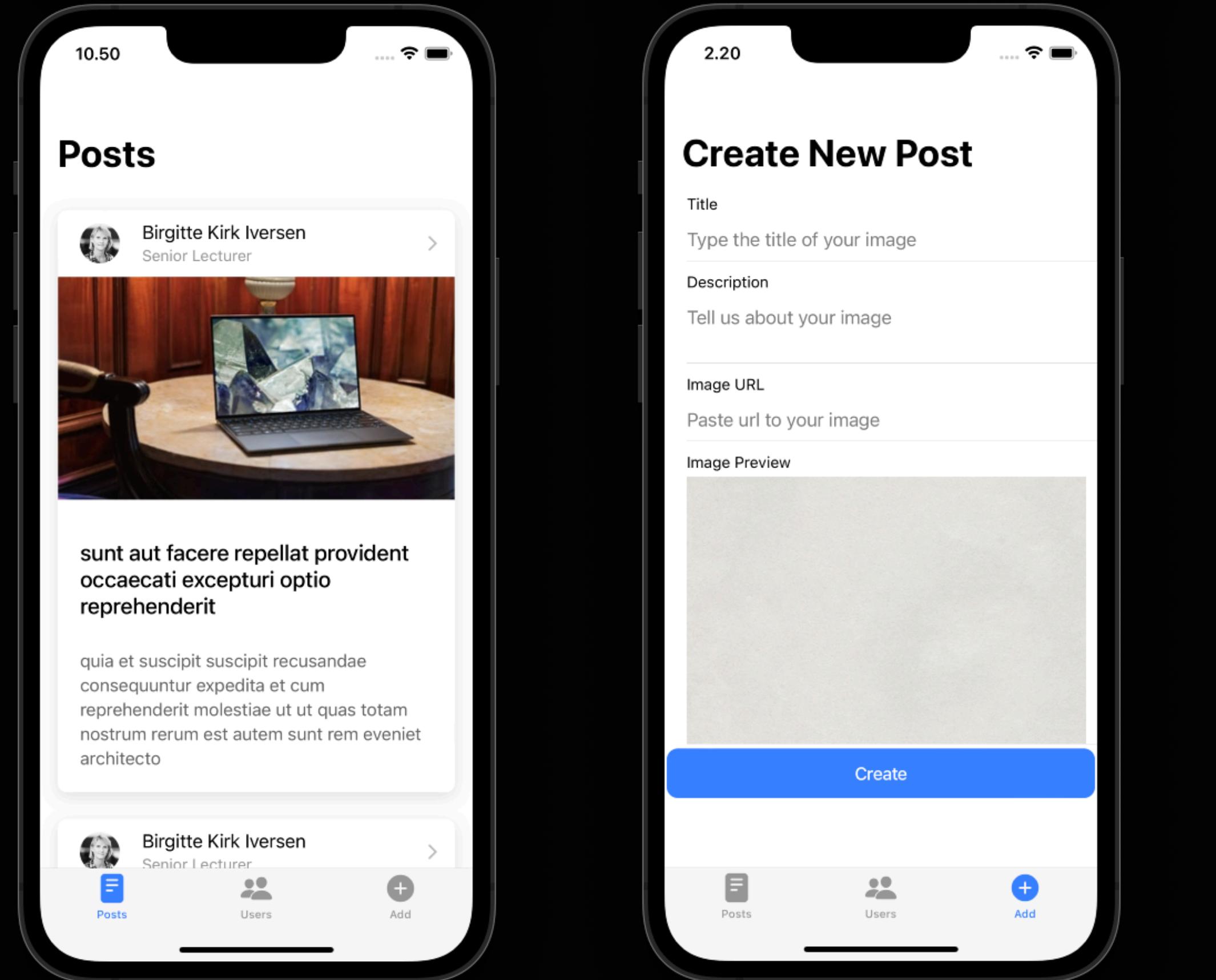
Deploy to iOS

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



Live Reload

<https://ionicframework.com/docs/react/your-first-app/live-reload>



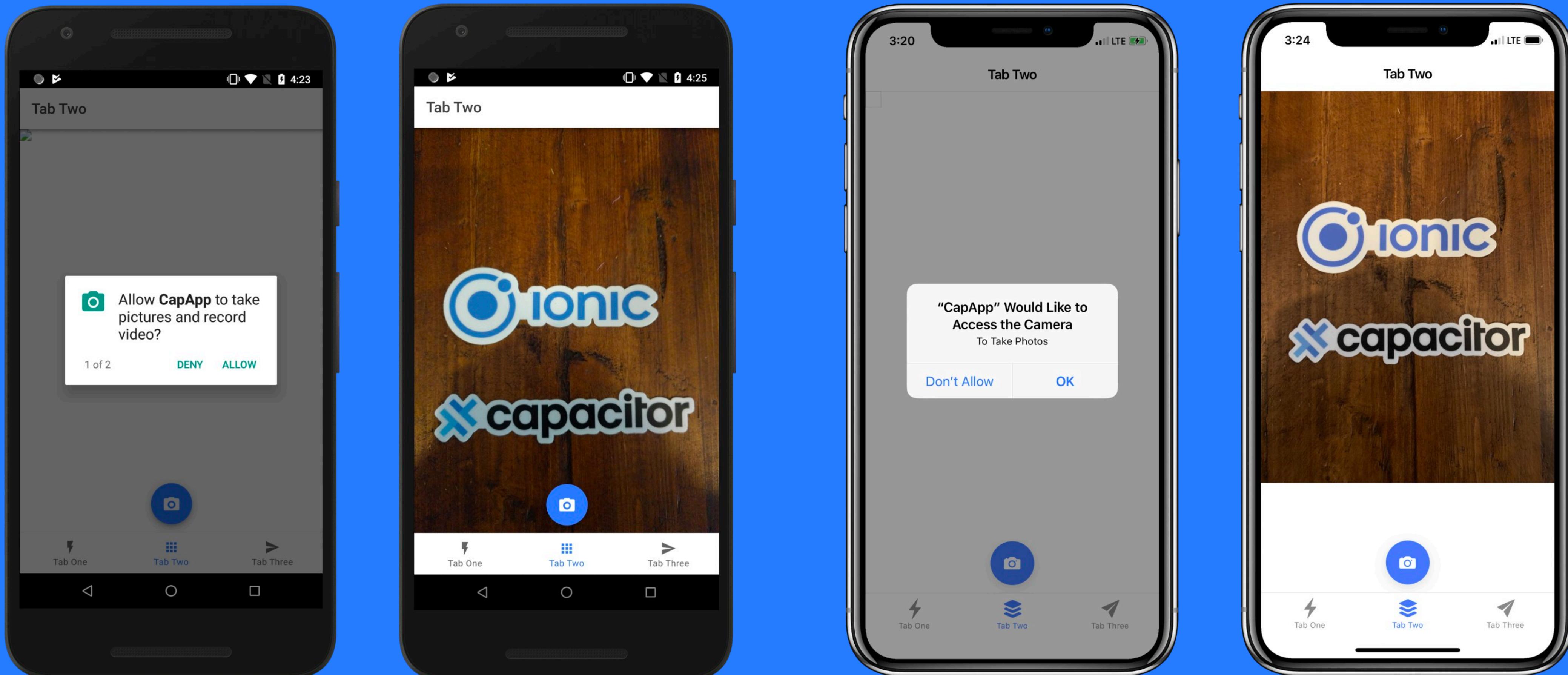
Deploy your Ionic Post App

Guides: [Android](#) & [iOS](#)

1. \$ ionic build
2. \$ ionic cap add ios or
ionic cap add android
3. Run on platforms with
Android Studio, Xcode or CLI.

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.'

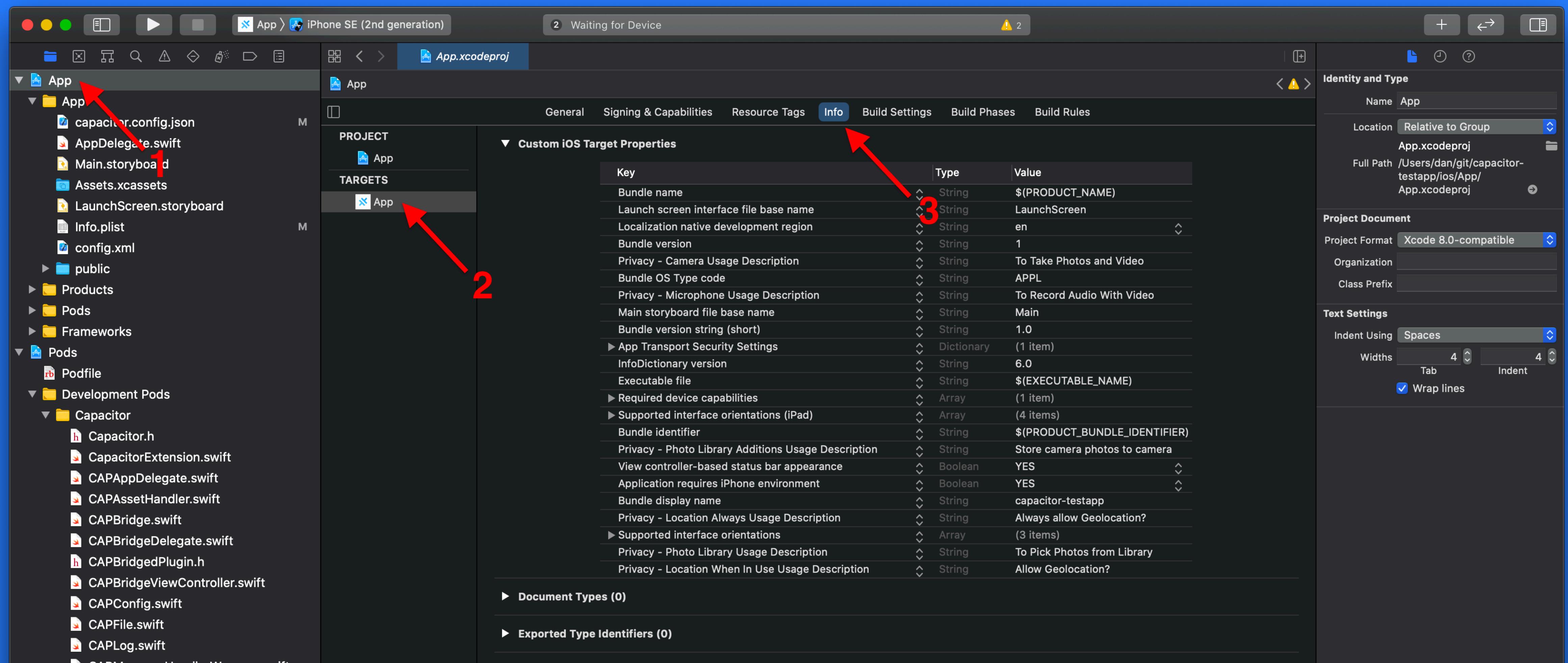
The left sidebar contains navigation links for 'Environment Setup', 'Installation', 'Using with Ionic Framework', 'Basics' (with sub-links for Development Workflow, Using Plugins, Native Project Configuration, Utilities), 'Upgrade Guides', 'Cordova/PhoneGap', 'Concepts', and 'iOS' (with sub-links for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

The top right features a search bar, navigation icons, and links for 'Docs', 'Plugins', 'CLI', 'Community', 'Blog', 'Enterprise', and social media sharing.

A sidebar on the right lists 'CONTENTS' including 'Configuring AndroidManifest.xml', 'Changing the Package ID', 'Changing the App Name', 'Deeplinks (aka Android App Links)', 'URL Schemes', and 'Setting Permissions'. There is also 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>

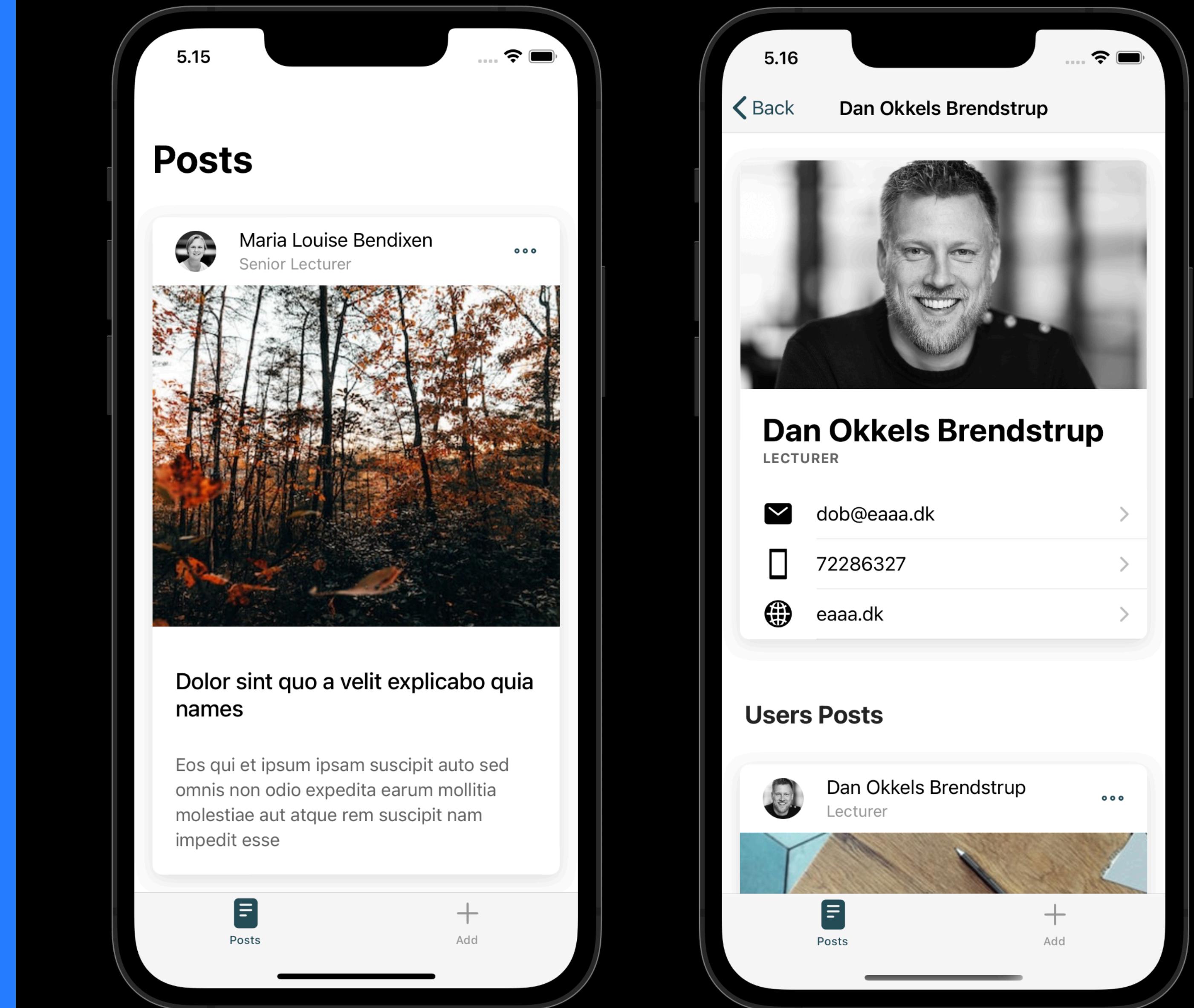


Native Device Features & Plugins

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

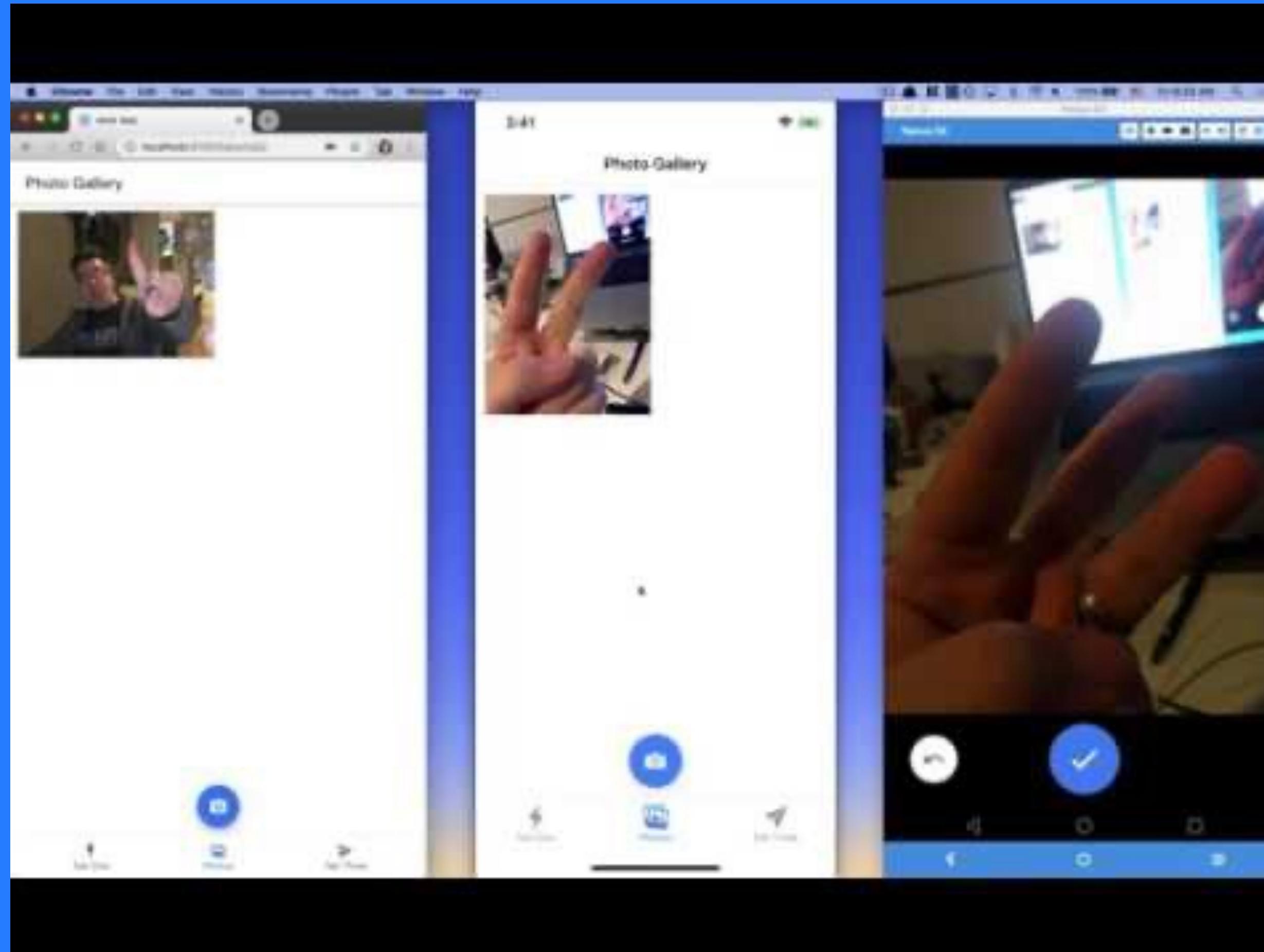
Ionic Post App w/ Firebase REST

Step 6 - 10



Ionic Camera App

<https://ionicframework.com/docs/react/your-first-app#create-an-app>



Ionic Post App w/ @capacitor/camera API

```
import { Camera, CameraResultType } from '@capacitor/camera';

const takePicture = async () => {
  const image = await Camera.getPhoto({
    quality: 90,
    allowEditing: true,
    resultType: CameraResultType.Uri
  });

  // image.webPath will contain a path that can be set as an image src.
  // You can access the original file using image.path, which can be
  // passed to the Filesystem API to read the raw data of the image,
  // if desired (or pass resultType: CameraResultType.Base64 to getPhoto)
  var imageUrl = image.webPath;

  // Can be set to the src of an image now
  imageElement.src = imageUrl;
};
```

1. On the Add Page, replace the URL input with the ability to take a photo with the camera or choose an existing one from the photo album.
2. Explore the @capacitor/camera and get inspired by the Ionic Camera App as well.
3. Install the @capacitor/camera.
4. Use CameraResultType.Base64 and save the output in the image property.

```
import { Storage } from '@capacitor/storage';

const setName = async () => {
  await Storage.set({
    key: 'name',
    value: 'Max',
  });
};

const checkName = async () => {
  const { value } = await Storage.get({ key: 'name' });

  alert(`Hello ${value}!`);
};

const removeName = async () => {
  await Storage.remove({ key: 'name' });
};
```

Ionic Post App w/ @capacitor/storage

1. Explore and install the @capacitor/storage plugin.
2. Reimplement the services with @capacitor/storage. Use the docs to explore how to get and set data.

Ionic App Project Description x +

https://race.notion.site/Ionic-App-Project-Description-6dc82eb47b30480c908e40c141326426 Incognito

MAD - Ionic React / Ionic App Project Description Search Duplicate ... Try Notion

Ionic App Project Description

Develop a Mobile App based on themes, concept and terms from the first part of the Mobile App Dev Course.

The Mobile App is developed in groups of 2-3 students. The project is based on a self-chosen theme and must meet the below technical requirements, demonstrating your learned skills and competencies from the course.

There are no requirements in addition to research and project management but feel free to use known project tools. However the main purpose is to practice programming and development of mobile apps with React, Ionic React and Capacitor JS.

The project is finalised by handing in a link to a GitHub repository including a README.md file with a short description of your app.

[Ionic App Project Description](#)

Next Tuesday

The screenshot shows a Notion page titled "MAD - Ionic React". On the left, there is a table titled "Course Overview" with columns for Date, Name, and Notes. The table lists ten entries from August 23 to October 18, 2022. The entry for "13/09/2022" is highlighted with a blue border. On the right, a detailed view of the "4. Ionic Project 1" section is shown, including fields for Lecturer (Empty), Date (13/09/2022), Notes (Group Work), and Course (MAD). Below this, a section titled "Group Work - Ionic Project" contains the instruction "Hand in: 21st of October, no later than 12.00 (noon)". A "Project Description" section includes a link to "Ionic App Project Description".

Course Overview

Date	Name	Notes
23/08/2022	1. Introduction to Mobile App Dev	Course Intro
30/08/2022	2. Web Native, Tools & UI Components	
06/09/2022	3. Native Device Features	Ionic Project Kick Off
13/09/2022	4. Ionic Project 1	Group Work
20/09/2022	5. Storage & Native APIs	Internship info at 14.00
27/09/2022	6. Firebase & App Dev	Final Ionic React lecture
04/10/2022	7. Ionic Project 2	Group Work
11/10/2022	8. Ionic Project 3	Group Work
18/10/2022	9. Ionic Project 4	Group Work

4. Ionic Project 1

Lecturer: Empty

Date: 13/09/2022

Notes: Group Work

Course: MAD

Group Work - Ionic Project

Hand in: 21st of October, no later than 12.00 (noon).

Project Description:

[Ionic App Project Description](#)