

Powered by  ionic

**It's time we talked about the
elephant in the room.**

**Hybrid mobile apps have been
built the same way for almost
10 years.**

**While great for accelerating
app development across
teams, the architecture
underneath is antiquated and
brittle**

<https://webnative.tech/>

-

Introduction to Mobile App Dev



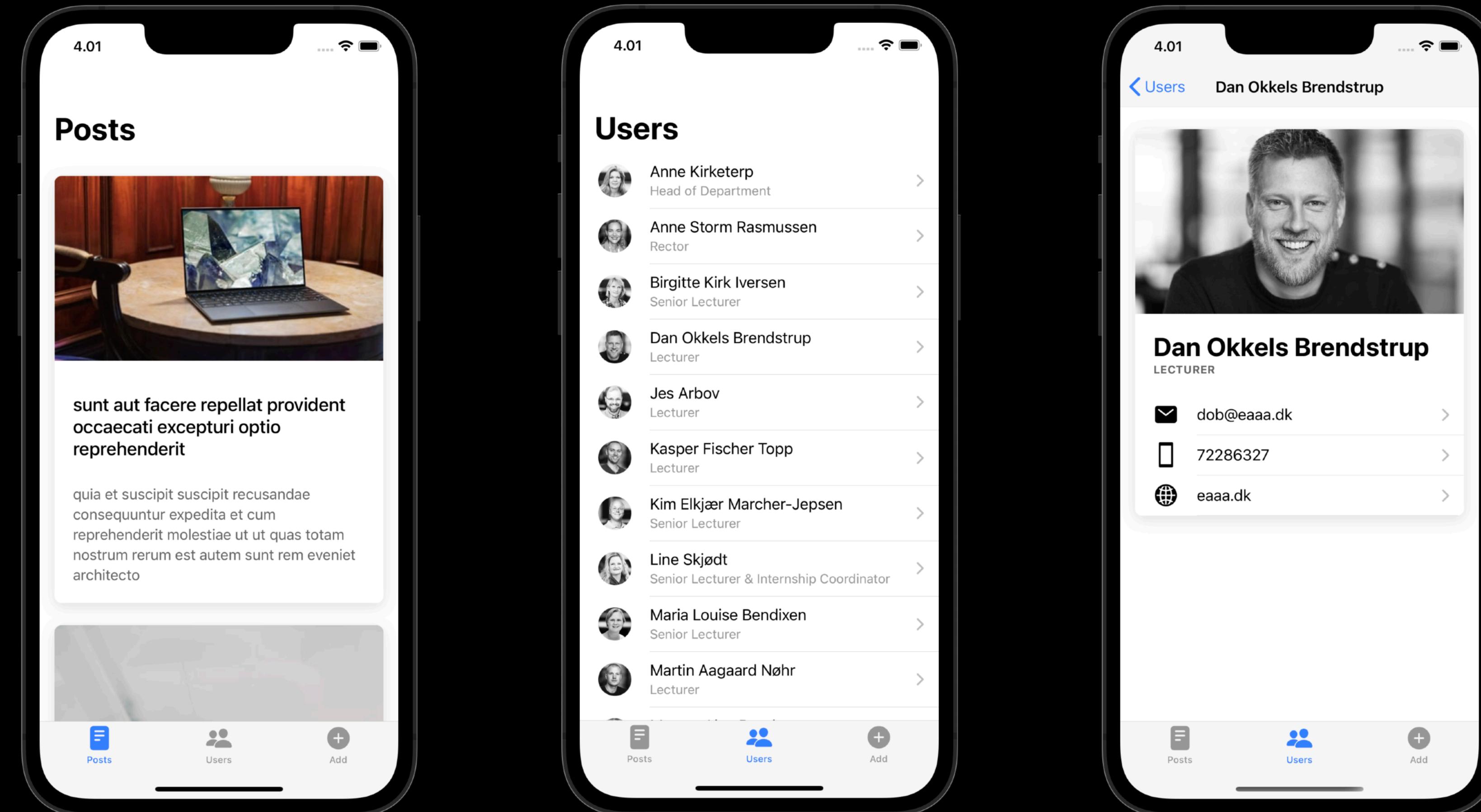


Agenda

- Course Overview & Practicalities
- What's mobile app dev?
- Types of Apps
- Ionic & Ionic React
- CLI & Tools
- Getting started w/ Ionic React



Demo



[ionic-post-app-firebase-rest-v6](#)

Mobile App Dev

... deals with programming techniques, user interface development and web integration which are necessary to develop applications for mobile devices - from idea to fully developed, distribution-ready application for one (or more) platform.

Uge 34				
Mandag d. 22 - 08	Tirsdag d. 23 - 08	Onsdag d. 24 - 08	Torsdag d. 25 - 08	Fredag d. 26 - 08
08:30 - 10:00	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
10:30 - 12:00	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
12:30 - 14:00	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
14:15 - 15:45				

Uge 35				
Mandag d. 29 - 08	Tirsdag d. 30 - 08	Onsdag d. 31 - 08	Torsdag d. 01 - 09	Fredag d. 02 - 09
08:30 - 10:00 Bemærk: AWP Project	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
10:30 - 12:00 Bemærk: AWP Project	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
12:30 - 14:00 Bemærk: AWP Project	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
14:15 - 15:45				

Uge 36				
Mandag d. 05 - 09	Tirsdag d. 06 - 09	Onsdag d. 07 - 09	Torsdag d. 08 - 09	Fredag d. 09 - 09
08:30 - 10:00 Bemærk: AWP Project	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
10:30 - 12:00 Bemærk: AWP Project	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
12:30 - 14:00 Bemærk: AWP Project	Mobile RACE eaa-R104-1.12	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
14:15 - 15:45				

Uge 37				
Mandag d. 12 - 09	Tirsdag d. 13 - 09	Onsdag d. 14 - 09	Torsdag d. 15 - 09	Fredag d. 16 - 09
08:30 - 10:00 Bemærk: Ionic Project	Mobile Bemærk: Ionic Project	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
10:30 - 12:00 Bemærk: Ionic Project	Mobile Bemærk: Ionic Project	Progressive Web Apps DOB eaa-R104-1.12		Advanced Web Programming DOB eaa-R104-1.12
Mobile	Mobile	Progressive Web Apps	Advanced Web Programming	

Course Overview - Part 1

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
18/08/2022	9. Ionic Project 4	Group Work

Curriculum

<https://www.baaa.dk/media/o2riq0gp/curriculum-web-development-elective-catalogue-february-2022.pdf>

Exam

The exam is an individual written exam in the form of a report with a maximum of 15 pages including images, which is prepared on the basis of a group project.

Two Mobile App Projects

During the Mobile App Dev Course, you will be introduced to two approaches of Mobile App Development (Ionic React and Unity), finalised by a project.

The projects forms the basis of your exam in Mobile App Dev and must be handed in, in order to pass the course and participate in the exam at the end of the semester.

Comparing the different approaches, will be a part of the final individual written exam. Read more about the exam in the curriculum.

Two Mobile App Project

Scope

- The exam is an individual written exam in the form of a report with a maximum of 15 pages including images, which is prepared based on either your Ionic App or Unity App group project.
- During the exam, you must review your previous project, and improve and extend your mobile app (Ionic or Unity).
- The technical requirements are equal to the listed in the project descriptions (Ionic App Project or Unity App Project)
- In the exam project, you must compare the two different approaches to mobile app development (Ionic VS Unity) and reflect on the pros and cons.

GitHub

The screenshot shows a GitHub repository page for the user 'cederdorff' with the repository name 'web-mobile-app-dev'. The page is in dark mode. The 'Code' tab is selected, displaying a list of 85 commits. The commits are organized into several branches, with the main branch having 1 commit. The commits are dated 6 months ago and involve changes to files like '_slides', 'data', 'ionic-camera-app', 'ionic-geolocation', 'ionic-post-app-firebase-rest-v0', 'ionic-post-app-firebase-rest-v1', 'ionic-post-app-firebase-rest-v2', 'ionic-post-app-firebase-rest-v3', 'ionic-post-app-firebase-rest-v4', 'ionic-post-app-firebase-rest-v5', 'ionic-post-app-firebase-rest-v6', 'ionic-post-app-realtime-db-storage', 'ionic-post-app-realtime-db-storage', 'ionic-post-app-realtime-db', 'ionic-post-app-v2', and 'ionic-post-app-v3'. The repository has 1 star, 1 watching, and 4 forks. It also has no releases or packages published. The Languages section shows TypeScript (40.2%), JavaScript (32.6%), CSS (24.1%), and HTML (3.1%).

cederdorff / web-mobile-app-dev Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

cederdorff Button disabled when new post not valid 33a1bdc on 4 Mar 85 commits

_slides Create 05-Firebase&AppDev.pdf 6 months ago

data new project templated added 6 months ago

ionic-camera-app camera app added 6 months ago

ionic-geolocation Update Home.jsx 6 months ago

ionic-post-app-firebase-rest-v0 New post app templates added 6 months ago

ionic-post-app-firebase-rest-v1 uid from number to string 6 months ago

ionic-post-app-firebase-rest-v2 uid from number to string 6 months ago

ionic-post-app-firebase-rest-v3 uid from number to string 6 months ago

ionic-post-app-firebase-rest-v4 uid from number to string 6 months ago

ionic-post-app-firebase-rest-v5 uid from number to string 6 months ago

ionic-post-app-firebase-rest-v6 Clean up and validation. 6 months ago

ionic-post-app-realtime-db-storage Button disabled when new post not valid 6 months ago

ionic-post-app-realtime-db-storage Clean up 6 months ago

ionic-post-app-realtime-db Update PostsPage.jsx 6 months ago

ionic-post-app-v2 v2 added 6 months ago

ionic-post-app-v3 Update UserPage.jsx 6 months ago

About

No description, website, or topics provided.

Readme

1 star

1 watching

4 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

TypeScript 40.2% JavaScript 32.6%

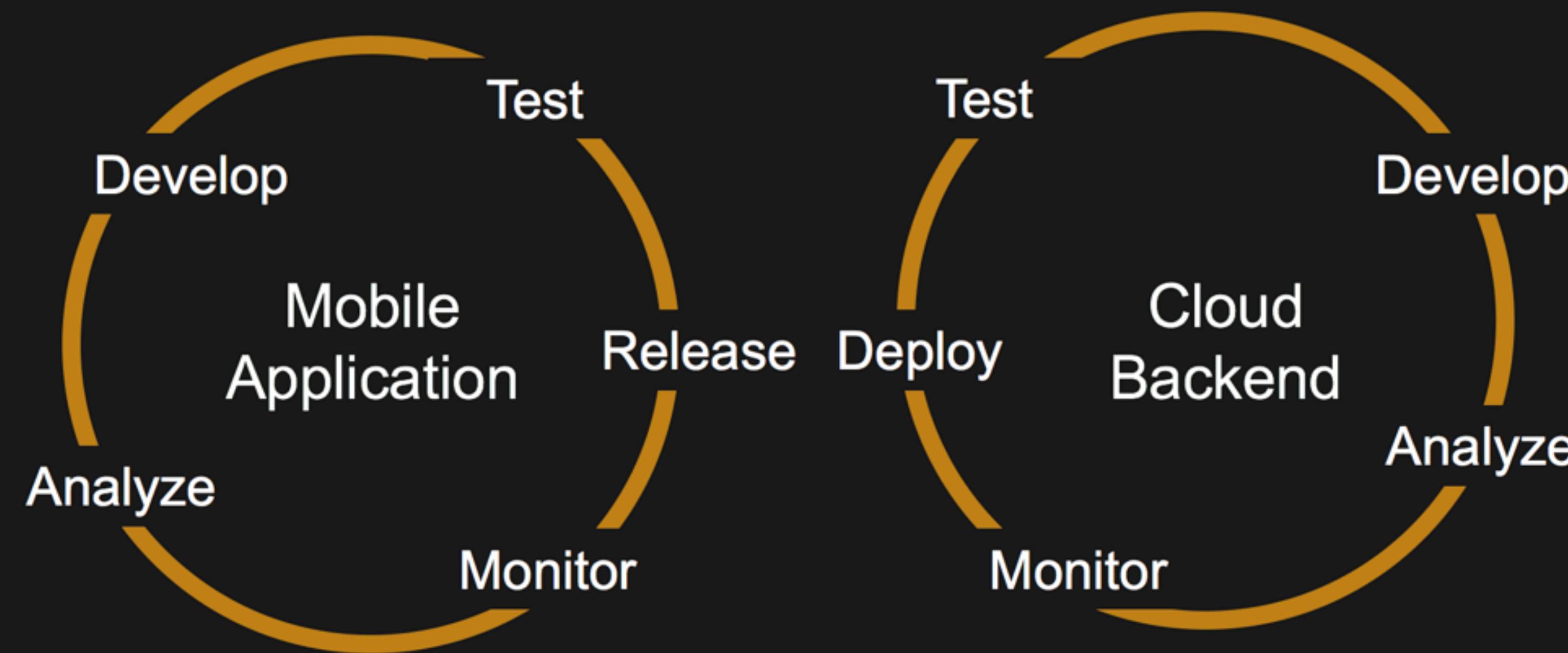
CSS 24.1% HTML 3.1%

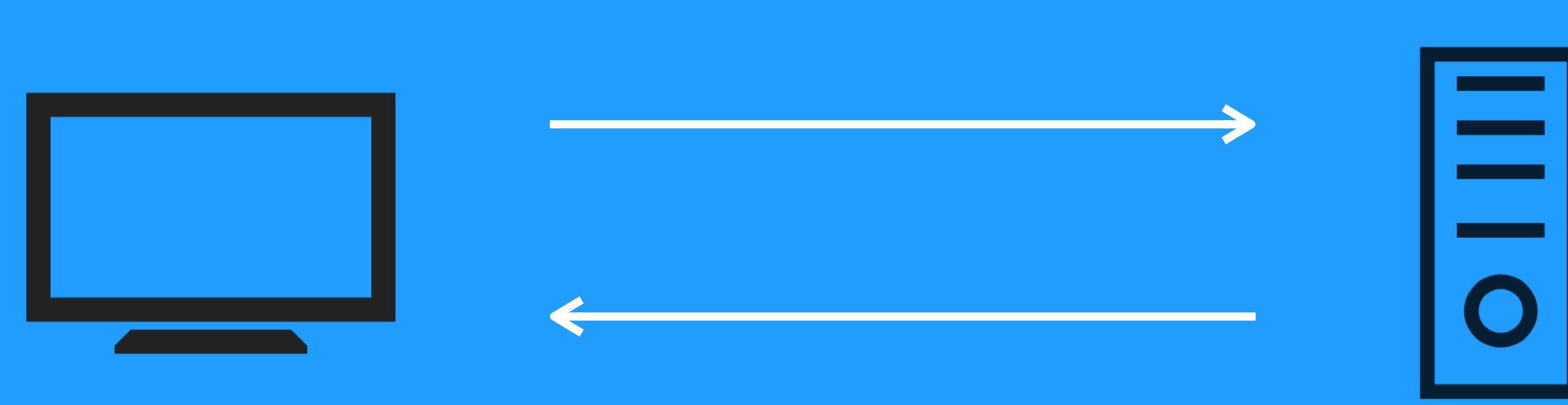
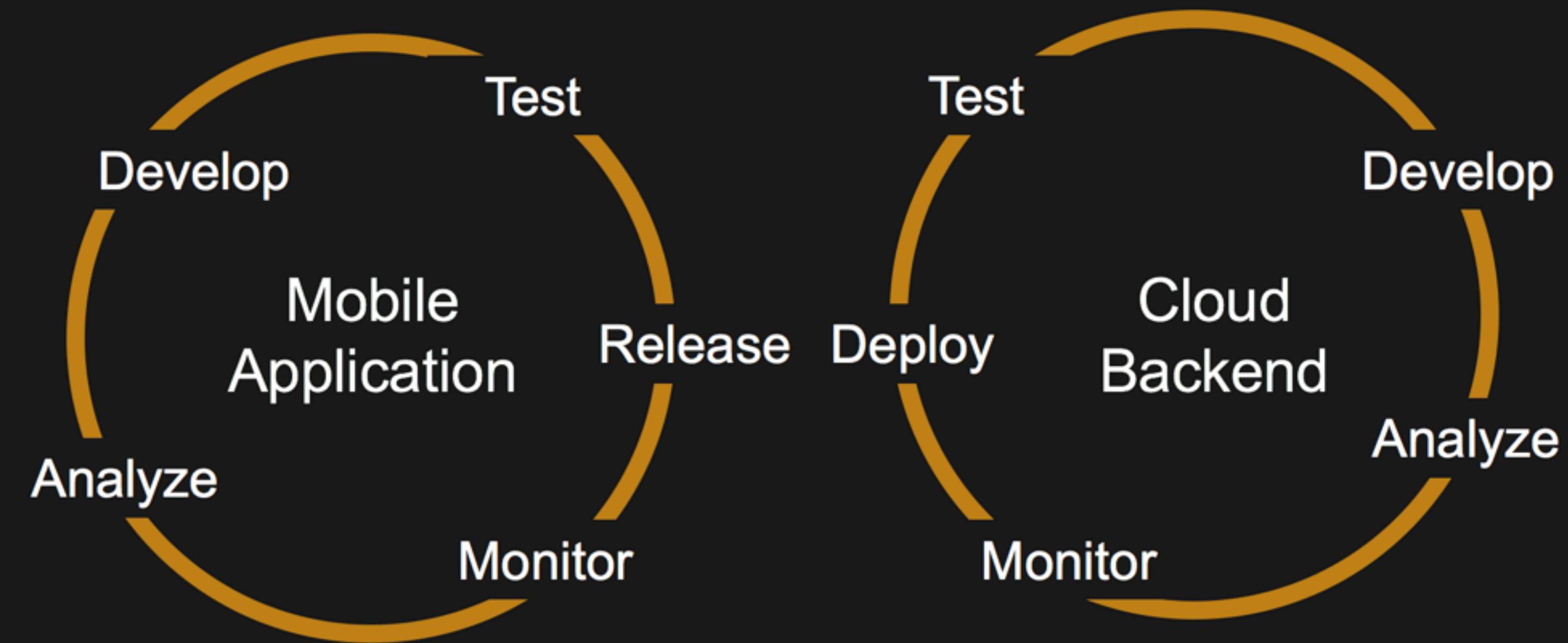
<https://github.com/cederdorff/web-mobile-app-dev>

What's mobile app dev?

The process of developing software applications that run on a mobile device, such as smartphones, tablets & smartwatches.

Mobile Dev Lifecycle

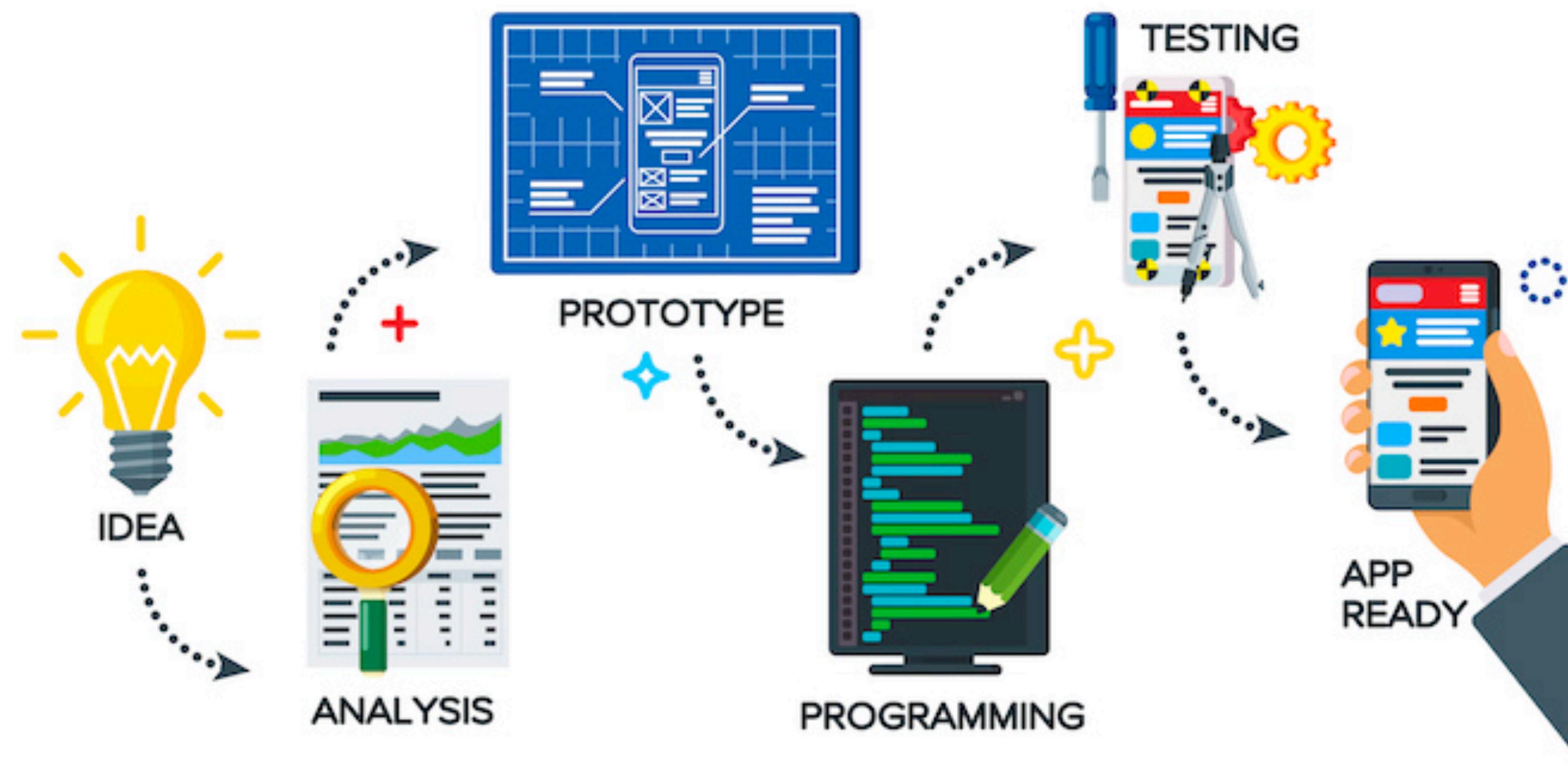




CLIENT

SERVER

“from idea to fully developed, distribution-ready application”



Download: App Store & Google Play



Native, Hybrid, Web Native & Web App

Types of Mobile Apps

Native



iOS

Android

Build with native & platform-specific languages & tools.

iOS - Xcode with Swift or Objective-C

Android - Android Studio with Kotlin or Java

Full and easier access to the device's capabilities.

“tend to also be more performant since their code is closer to the ‘metal’”.

Native user interface (UI) controls and layouts.

Android Apps cannot run on iOS and vice versa.

Hybrid

“Hybrid apps are native apps. They’re downloaded from the platform’s app store or marketplace and offer the same native features, offline support, and hardware-based performance acceleration as any app built with a native SDK.”

A blend - both native & web.

Build with HTML, CSS & JS (+ framework) & cross-platform like Ionic, NativeScript, Xamarin, React Native.

With plugins: full access to native features.

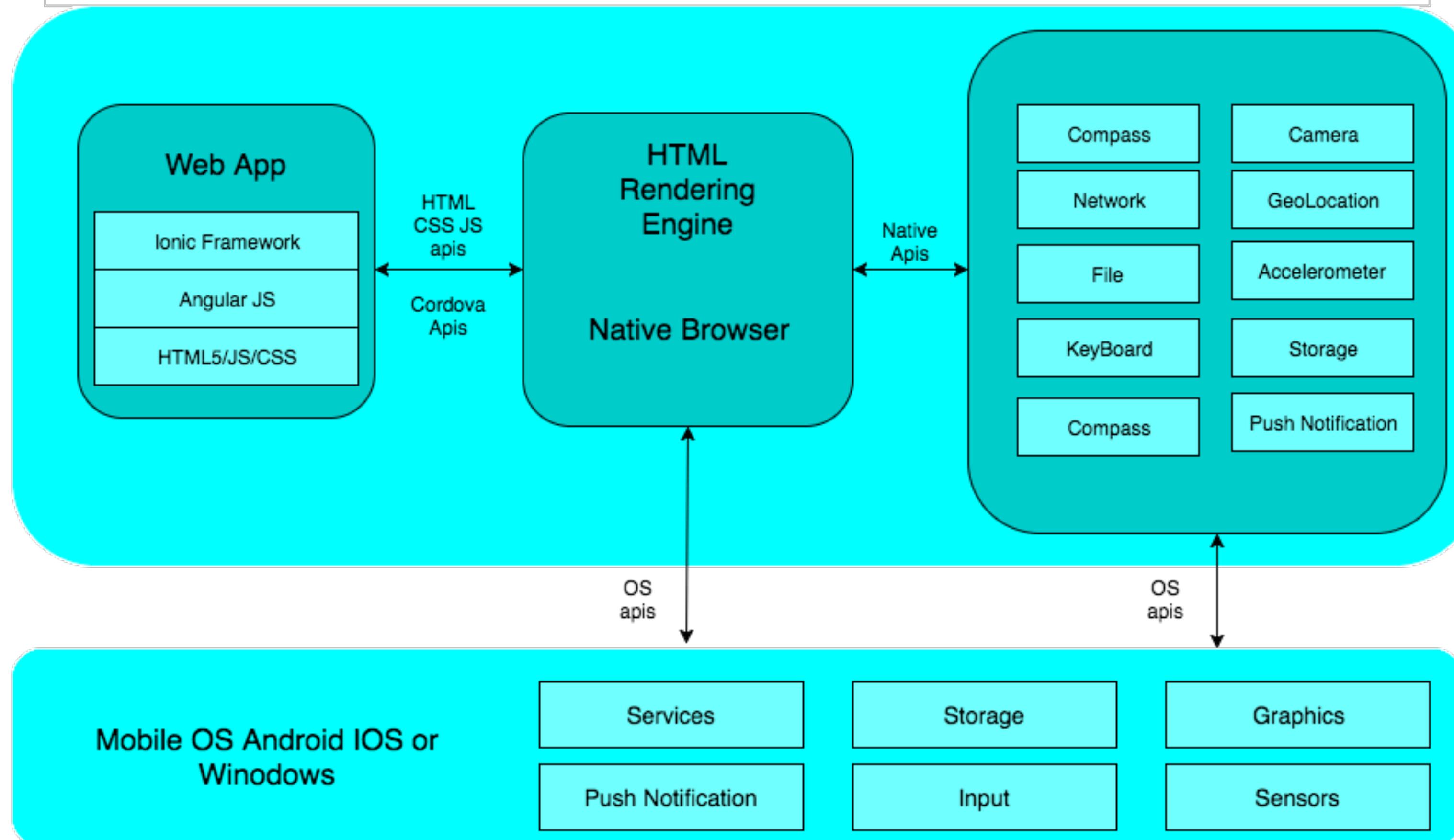
Embedded in WebView.

With a UI Framework: feels and looks like a pure native.

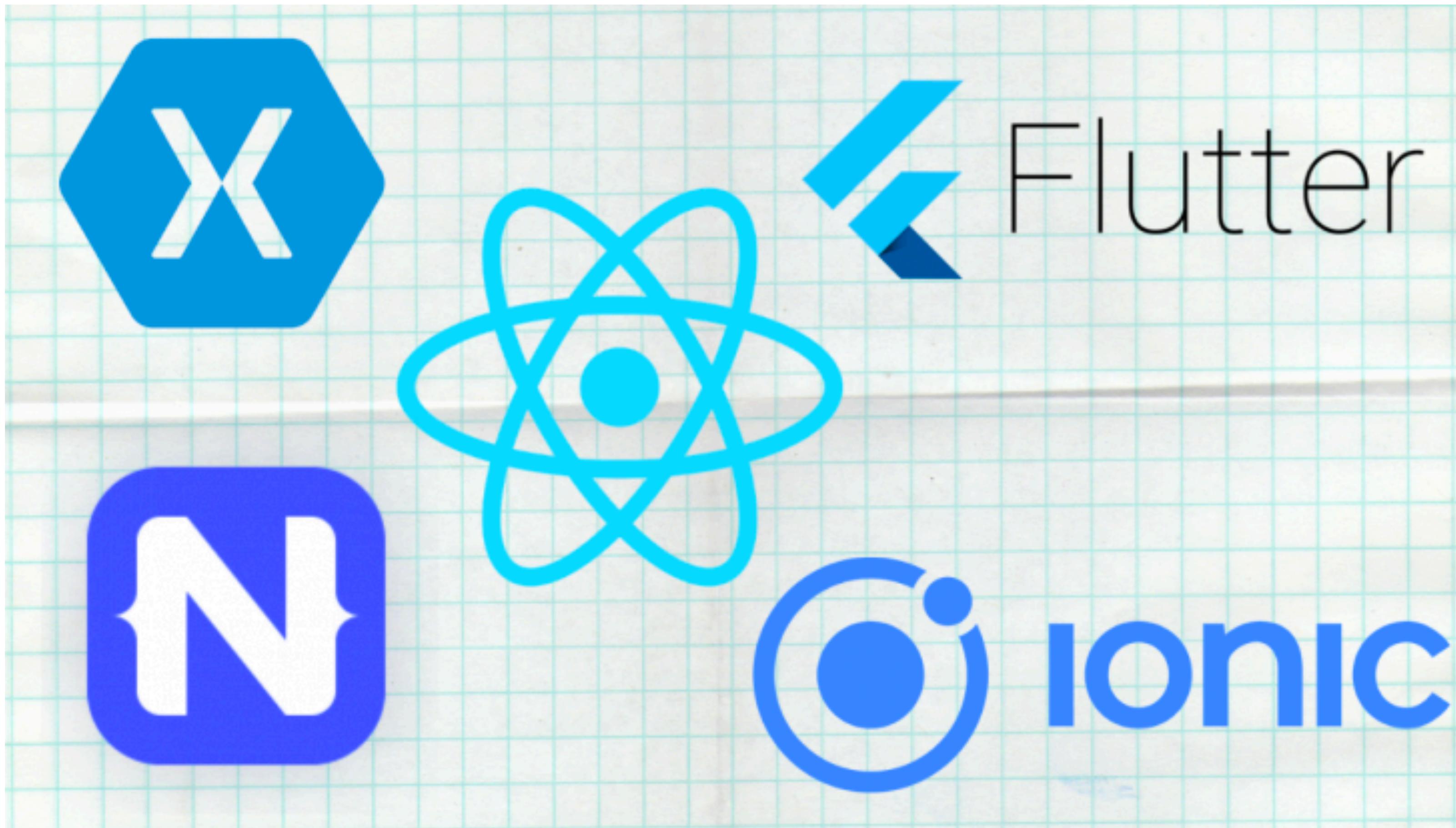
Bridge & plugins - Cordova & Capacitor.

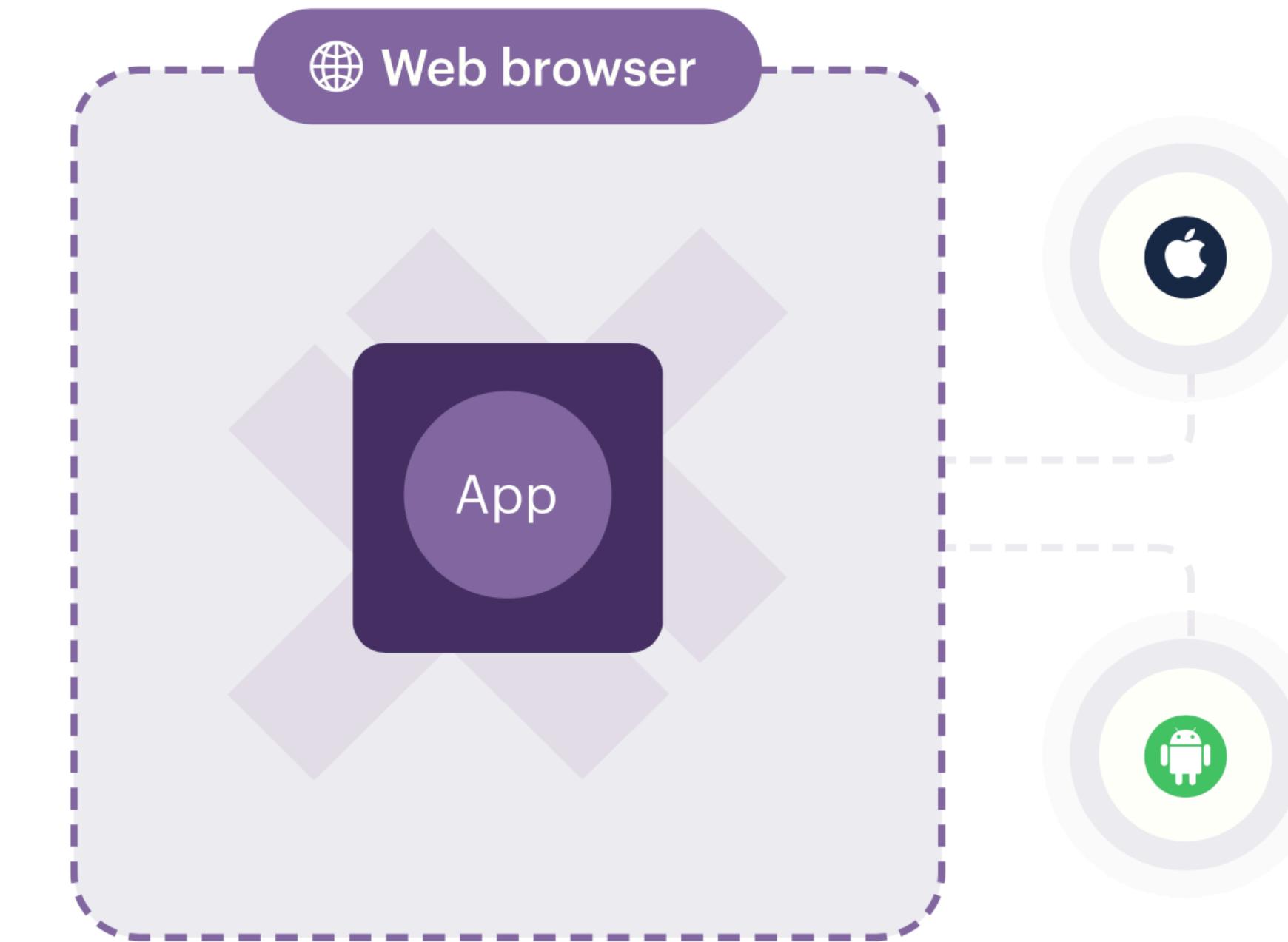
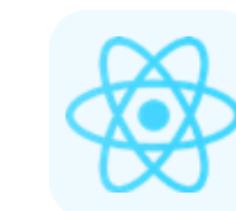


Hybrid App Architecture



Cross-platforms





Web App



Any program executed in the
browser.

Stored on a web server.

Accessible anywhere through a
URL.

Build with HTML, CSS & JS

TYPES OF WEB APPS

WEB BASED

BROWSER BASED

API BASED

SERVER APPS

PORTAL APPS

SOFTWARE AS A SERVICE

PROGRESSIVE WEB APP

SPA & MPA

Web Native

More than *just* Hybrid

Web Native

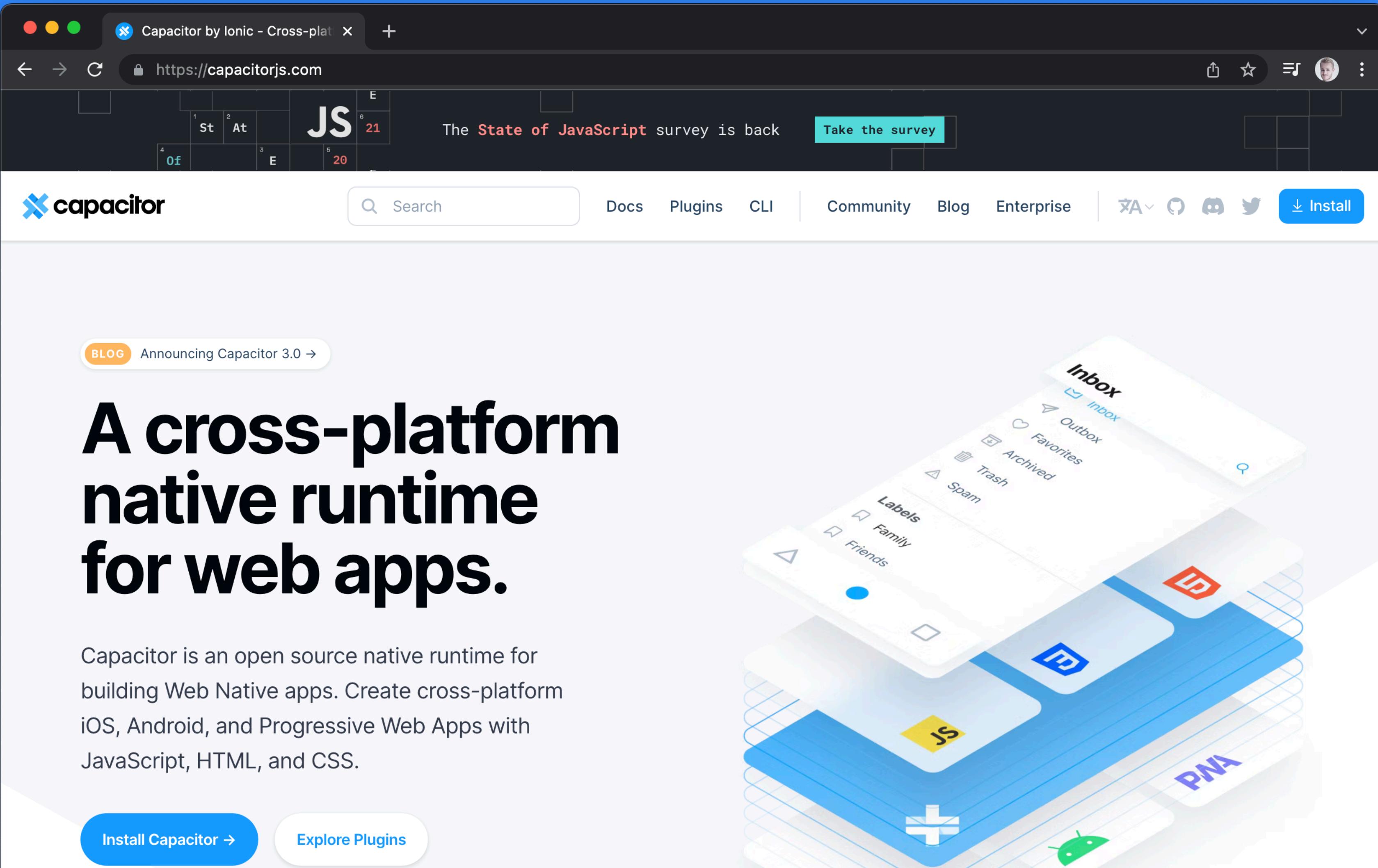
Is the full capability and access to developers of the Web Platform, with the full functionality and performance benefits of traditional native apps.

<https://webnative.tech/>

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

Capacitor

A cross-platform native runtime for web apps.



The screenshot shows a web browser displaying the Capacitor website at <https://capacitorjs.com>. The page features a large, bold headline: "A cross-platform native runtime for web apps.". Below the headline is a paragraph describing Capacitor as an open source native runtime for building Web Native apps. To the right of the text is a graphic of three smartphones showing different app interfaces: one with an "Inbox" screen, one with a "Labels" screen, and one with a "PNA" screen. At the bottom of the page are two call-to-action buttons: "Install Capacitor →" and "Explore Plugins". The browser's header includes the title "Capacitor by Ionic - Cross-plat", a search bar, and various browser controls.

BLOG Announcing Capacitor 3.0 →

A cross-platform native runtime for web apps.

Capacitor is an open source native runtime for building Web Native apps. Create cross-platform iOS, Android, and Progressive Web Apps with JavaScript, HTML, and CSS.

Install Capacitor → Explore Plugins

Back in the 2010s

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

Today

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

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.



Native vs Web Native (Hybrid)

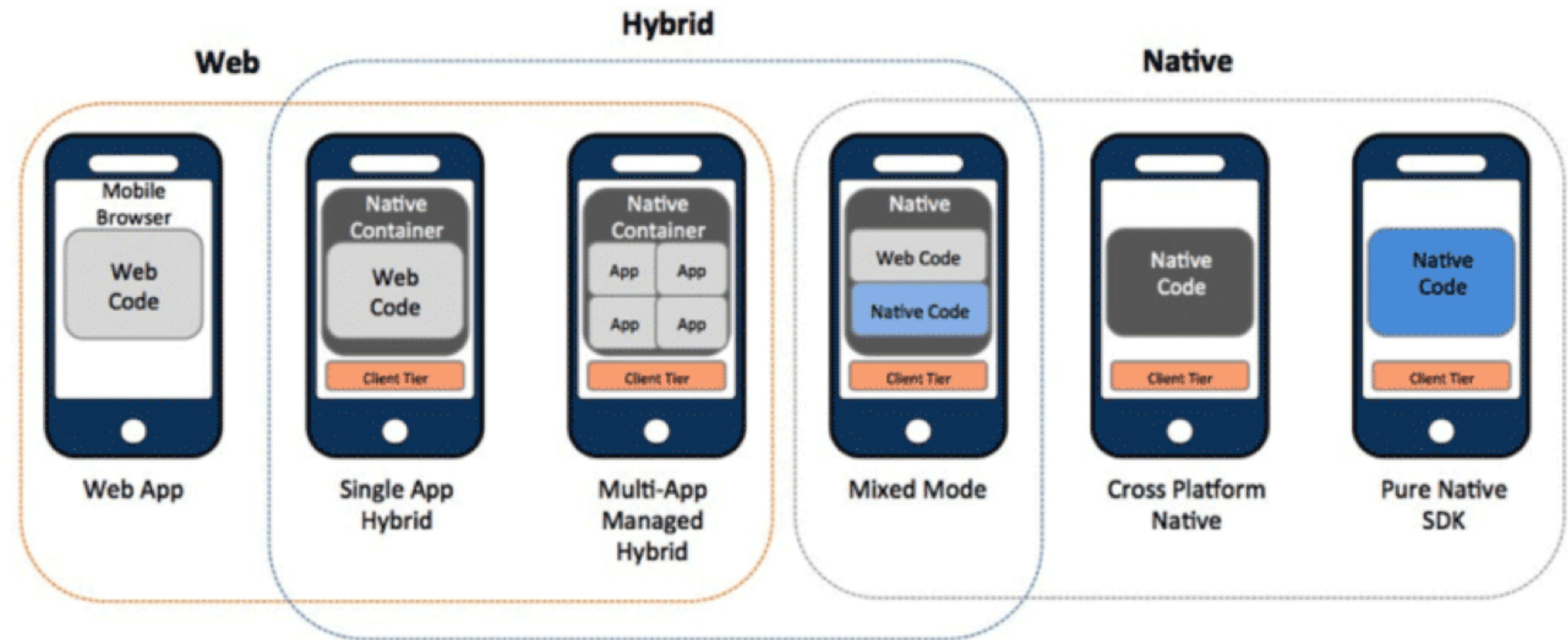
What are the differences? Pros & Cons?

“The key difference is that hybrid apps are built using open web technologies like HTML, CSS, and JavaScript, rather than the proprietary or specialized languages used by iOS, Android, and others. That means anyone with a modern web developer skill-set can begin building an app using the hybrid approach.

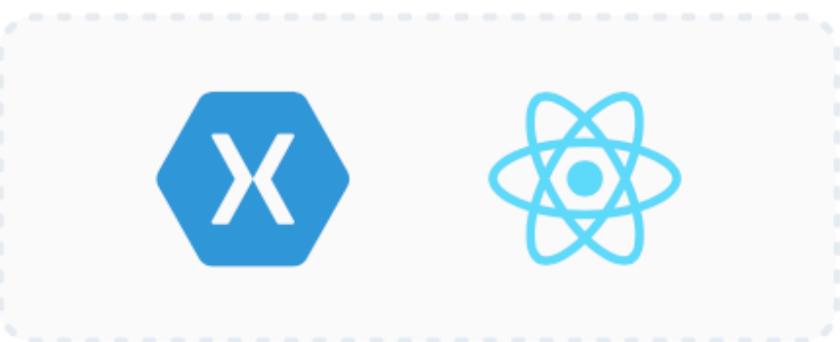
Hybrid apps run in a full-screen browser, called a webview, that is invisible to the user. Through customizable native plugins, they can access the native features of specific mobile devices (such as the camera or touch ID), without the core code being tied to that device.

That means cross-platform hybrid applications can run on any platform or device, all from a single codebase, while still delivering native performance.“





Xamarin/React Native



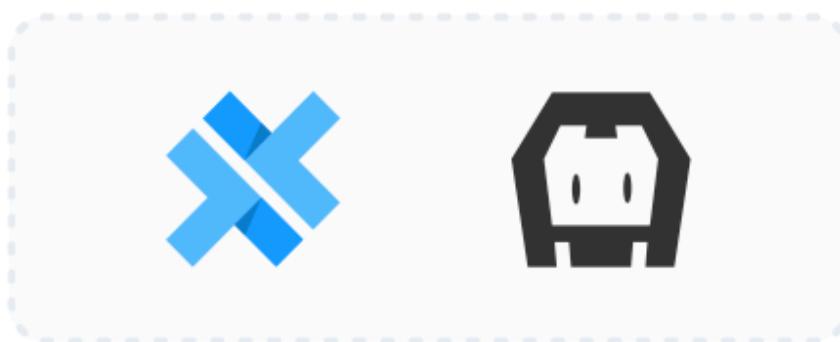
Native UI Controls

Native Runtime

Native APIs

Device OS

Capacitor/Cordova

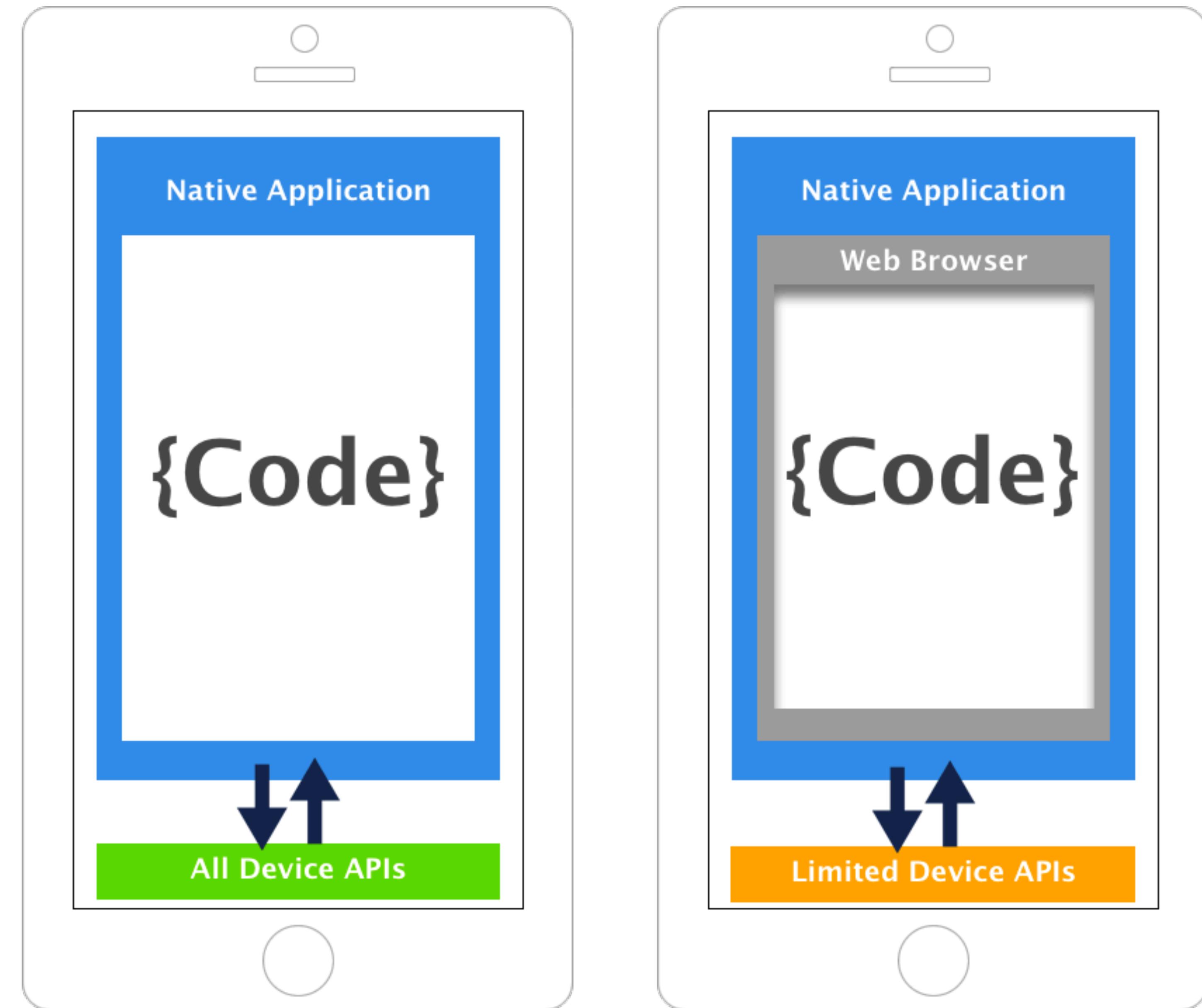


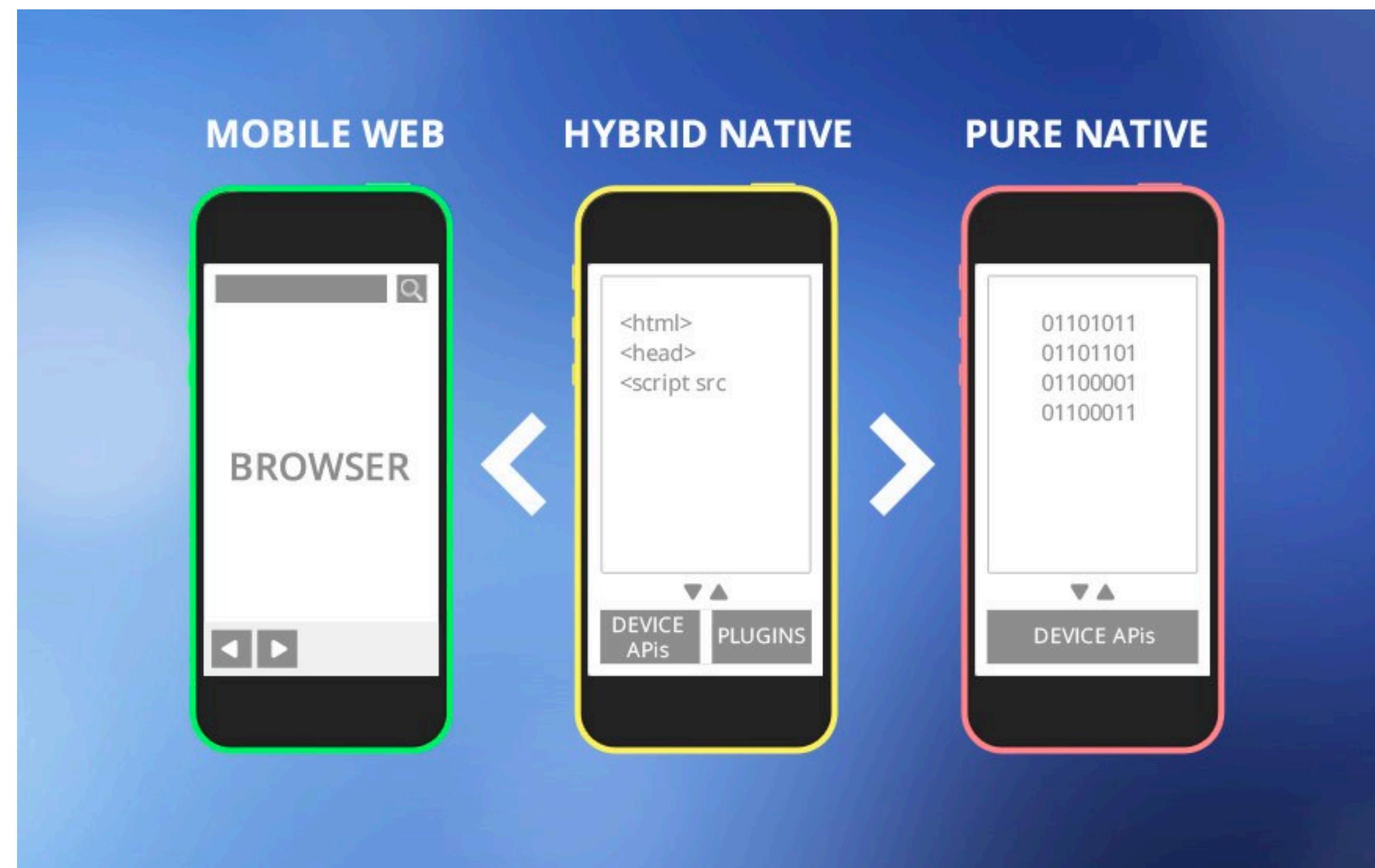
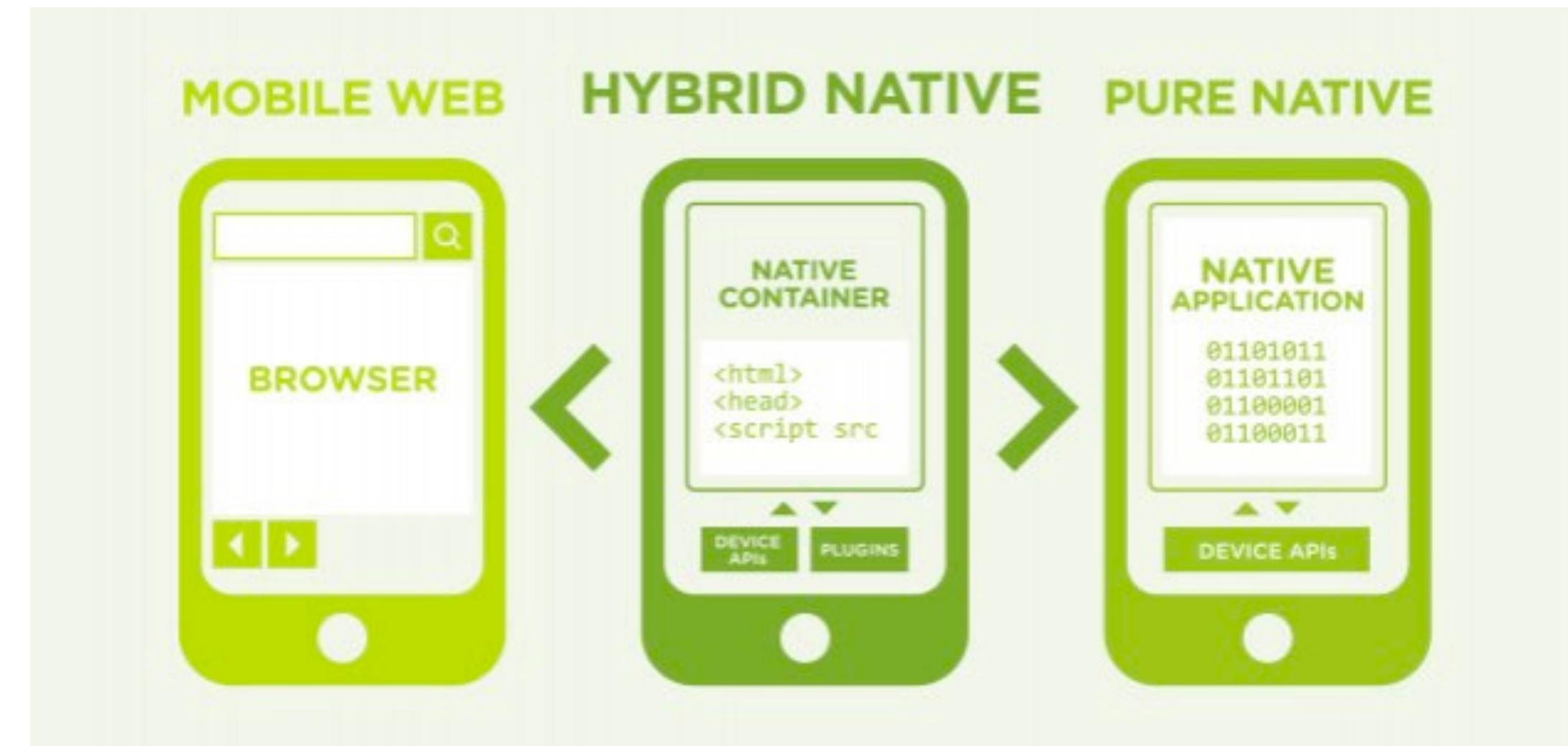
Web UI Controls

Native Runtime

Native APIs

Device OS





Web App

Hybrid App

Native App

HYBRID APPS



NATIVE APPS



Why Native? Why Hybrid?

Hybrid vs Native (ebook): <http://go.ionic.io/hybrid-vs-native-guide>

Why hybrid?

Let's take a look at each of these.



Write once, run anywhere



Use the talent you
already have



Deliver a great user
experience across
platforms



Build for the future

Traditional Approach



Ionic (Web Native) Approach





The Ionic logo features a large, white, sans-serif font spelling "ionic". To the left of the "i", there is a circular icon composed of two concentric arcs. The background is a vibrant blue with abstract white shapes, including diagonal stripes and small circles.

ionic

JS

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

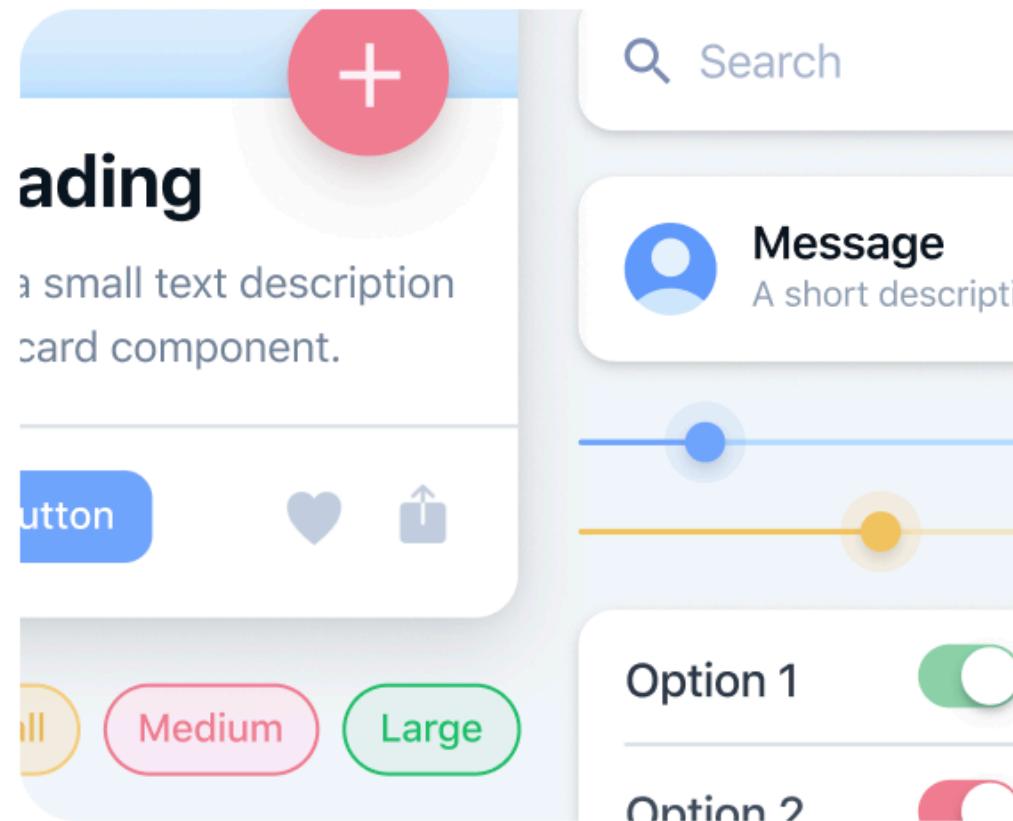
Ionic is a Cross-platform

And framework that offers mobile-optimized UI components, gestures,
and tools for build apps (iOS, Android, Web, PWA & Desktop)

Ionic is a Cross-platform

Yes, Hybrid (or Web Native)

Component library for building apps that run on
iOS, Android, Electron, and the Web.



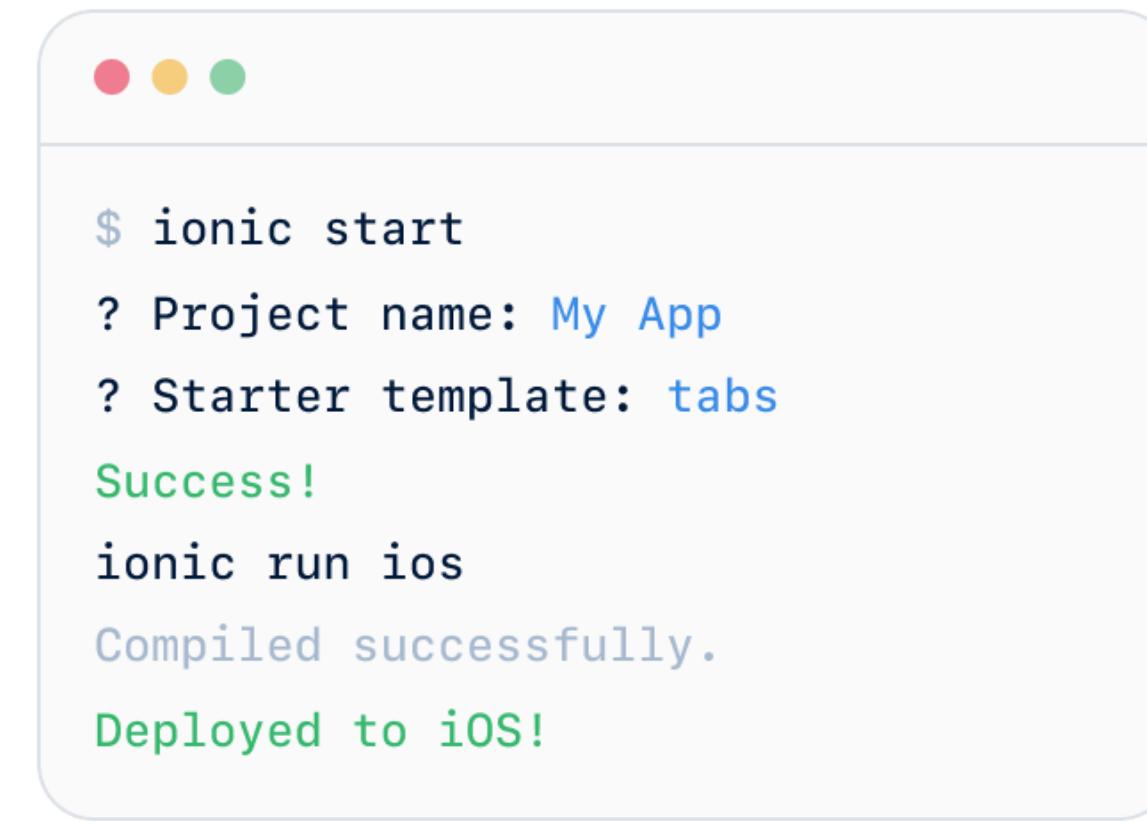
Pre-designed UI components

Ionic's [UI components](#) look great on all mobile devices and platforms. Start with pre-made components, typography, and a base theme that adapts to each platform.



Write once, run anywhere

Ionic lets developers to ship apps to the app stores and as a PWA with a single code base. With [Adaptive Styling](#), apps look and feel at home on every platform.



Developer-friendly tooling

Create, build, test, and deploy your app with the [Ionic CLI](#). Take advantage of Live Reload, deployments, integrations, and even use your favorite JS framework's CLI.

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 blue 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 sub-section titled "Action Sheet". Below this, there's a brief description of Ionic components and a list of available components:

- Action Sheet
- Accordion
- Alert
- Badge
- Breadcrumb
- Button
- Card
- Checkbox

Each component is represented by a small icon and a brief description. For example, the Action Sheet is described as a set of options with the ability to confirm or cancel an action.

Component	Description
Action Sheet	Action Sheets display a set of options with the ability to confirm or cancel an action.
Alert	Alerts are a great way to offer the user the ability to choose a specific action or list of actions.
Badge	Badges are a small component that typically communicate a numerical value to the user.
Card	Cards are a great way to display an important piece of content, and can contain images, buttons, text, and more.
Checkbox	Checkboxes can be used to let the user know they need to make a binary decision.
Chip	Chips are a compact way to display data or actions.
Content	Content is the quintessential way to interact with and navigate through an app.

**Build with whatever
you prefer**

Angular, React, Vue, or
Vanilla JavaScript



Ionic Framework



Installation Guide

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



Native Functionality

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



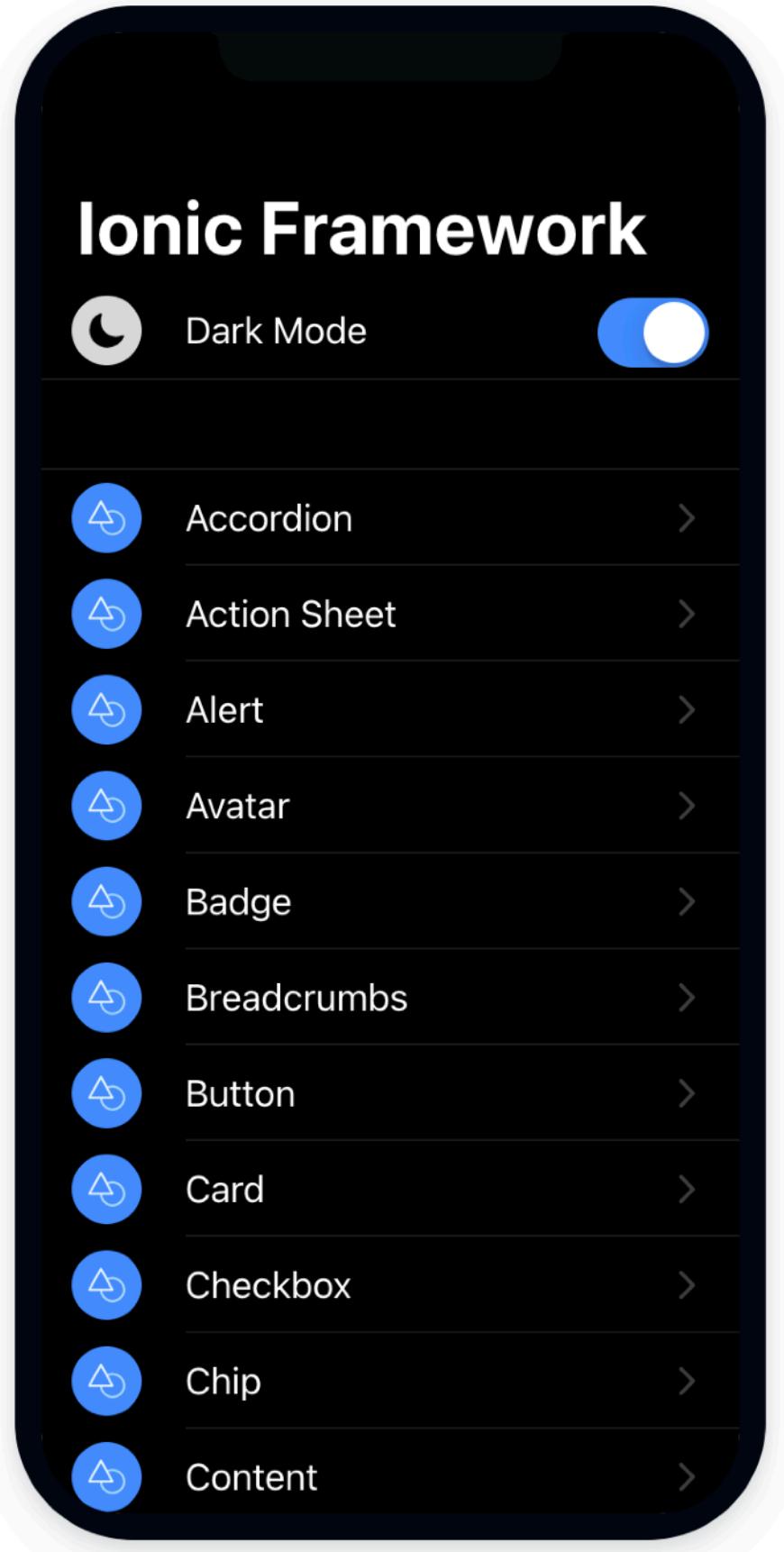
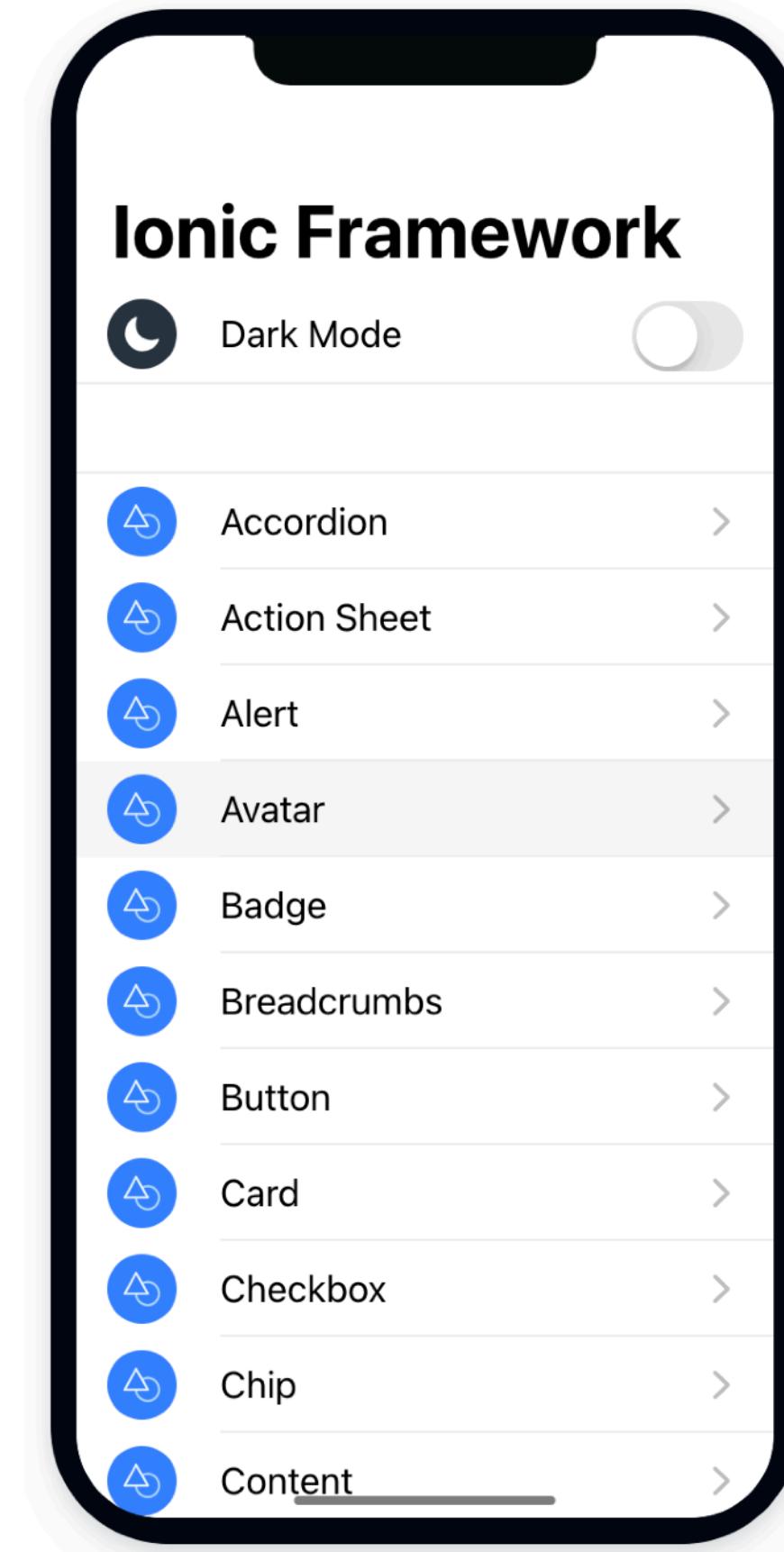
UI Components

Dive into Ionic beautifully designed UI component library.

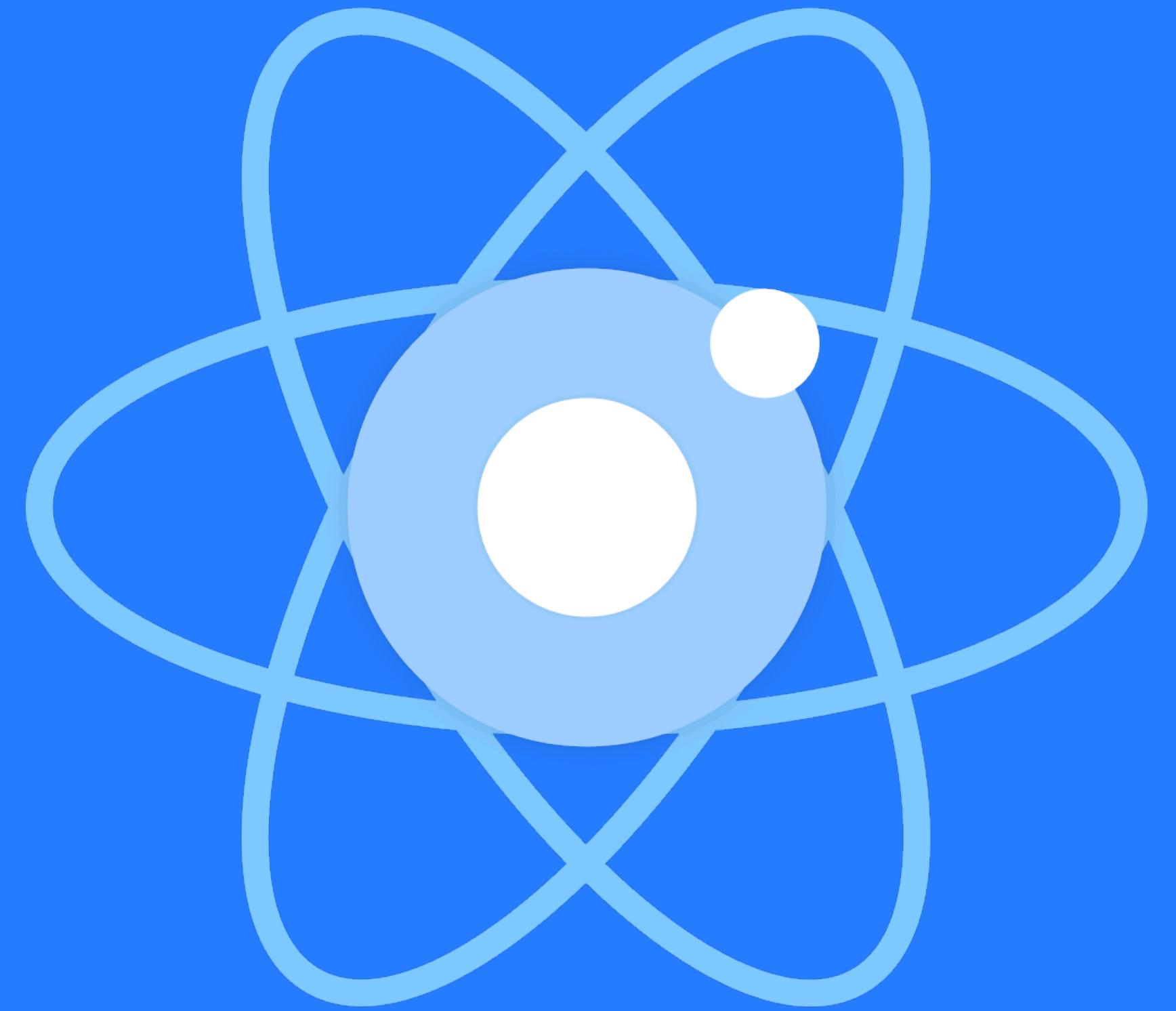


Theming

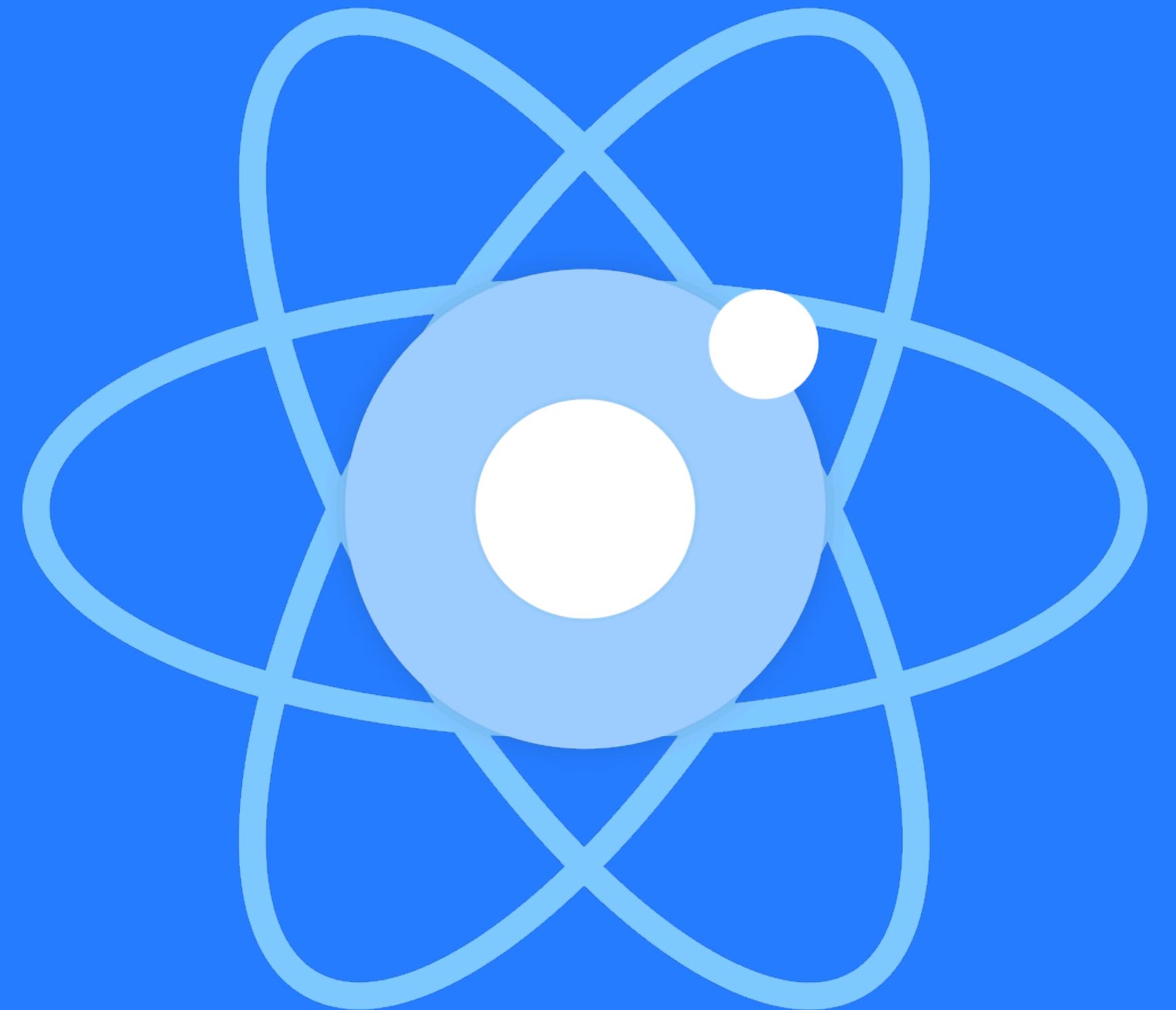
Learn to easily customize and modify your Ionic app's visual design to fit your brand.



<https://ionicframework.com/>



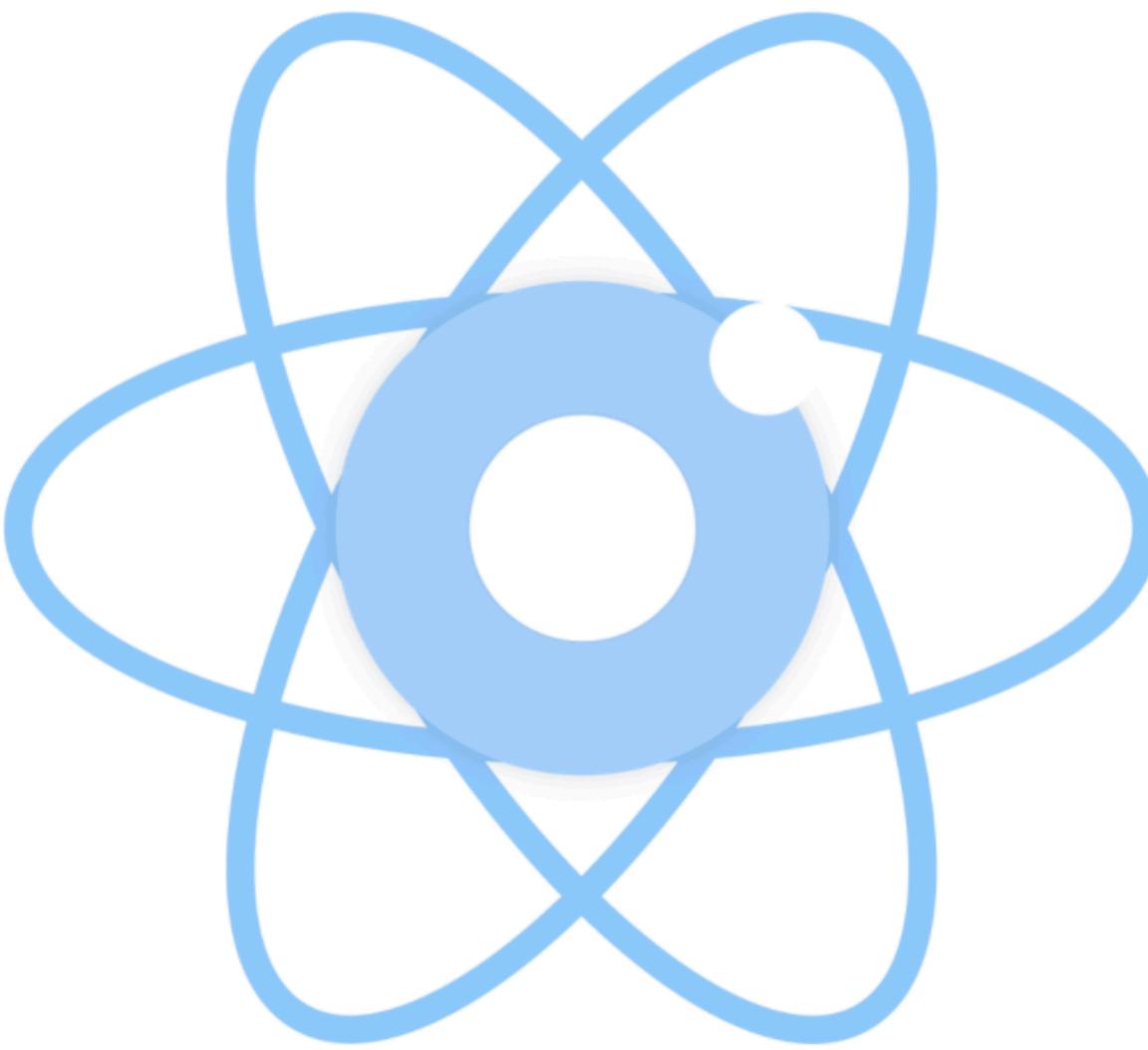
Ionic React



Build Native and
Progressive Web Apps
from a single code base
with Ionic React.

One Codebase Any Platform **Just React**

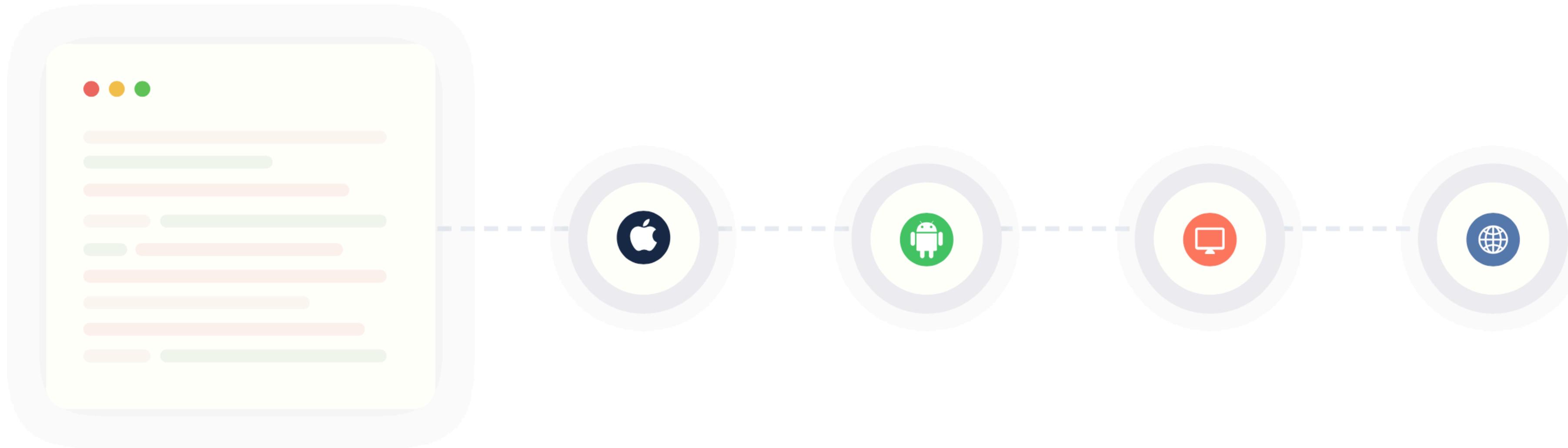
- ✓ 100+ mobile optimized React UI components
- ✓ Standard React tooling with react-dom
- ✓ iOS / Android / Electron / PWA



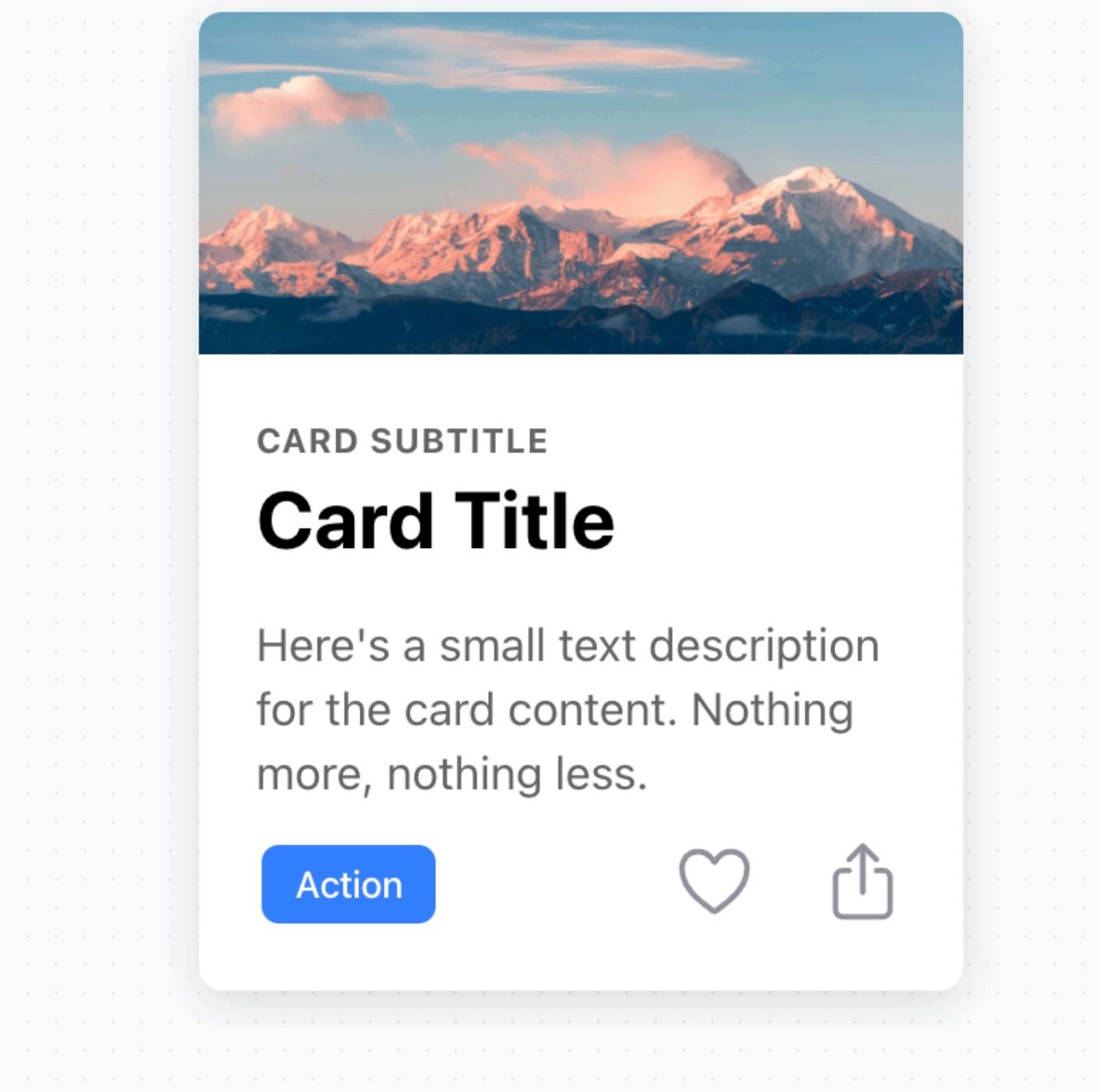
Why Ionic?

you
are
web developers!

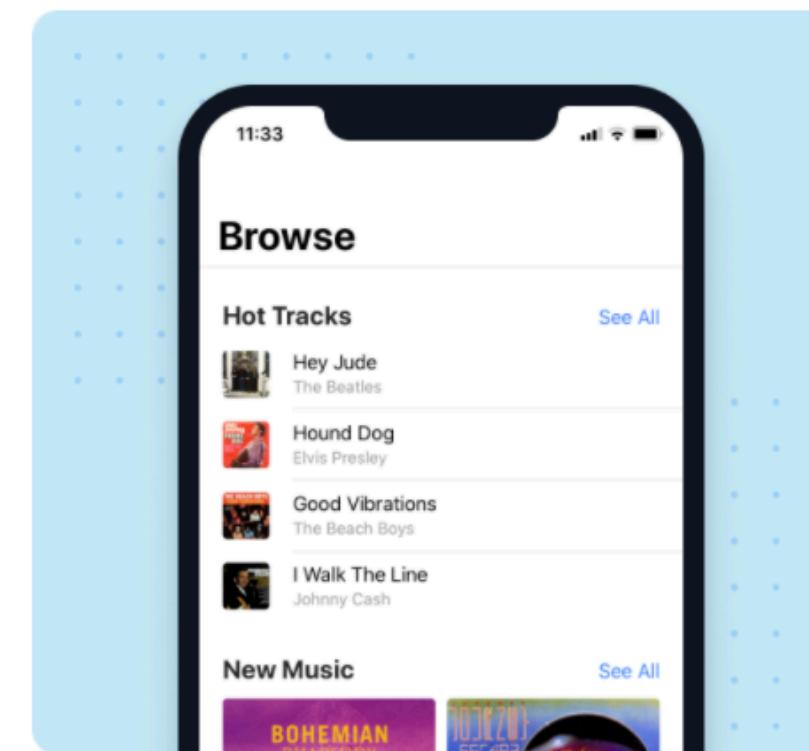
Seamless experience across platforms



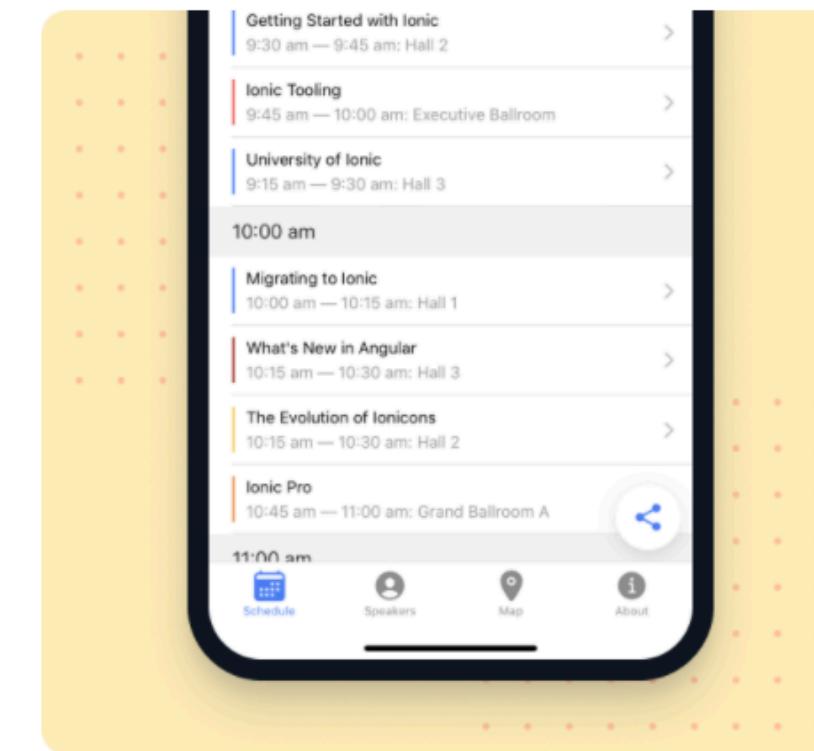
Predefined UI Components



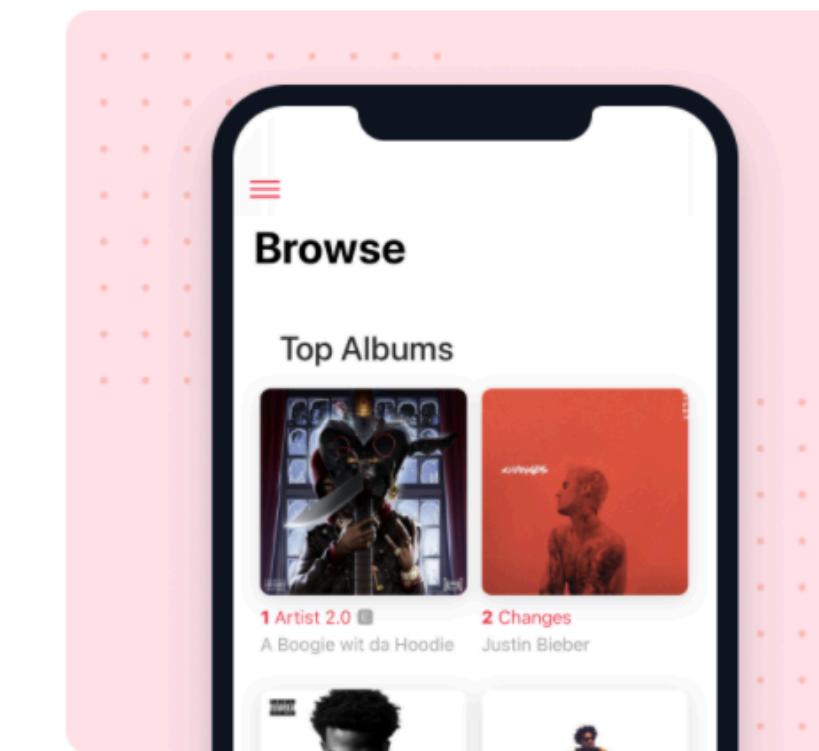
Awesome docs with many examples
and sample apps



Music Player



Conference App

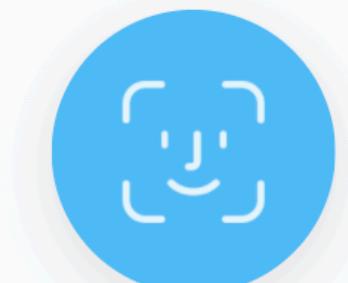


StarTrack

Integrations



Apple Pay



Integrate with any
backend tech

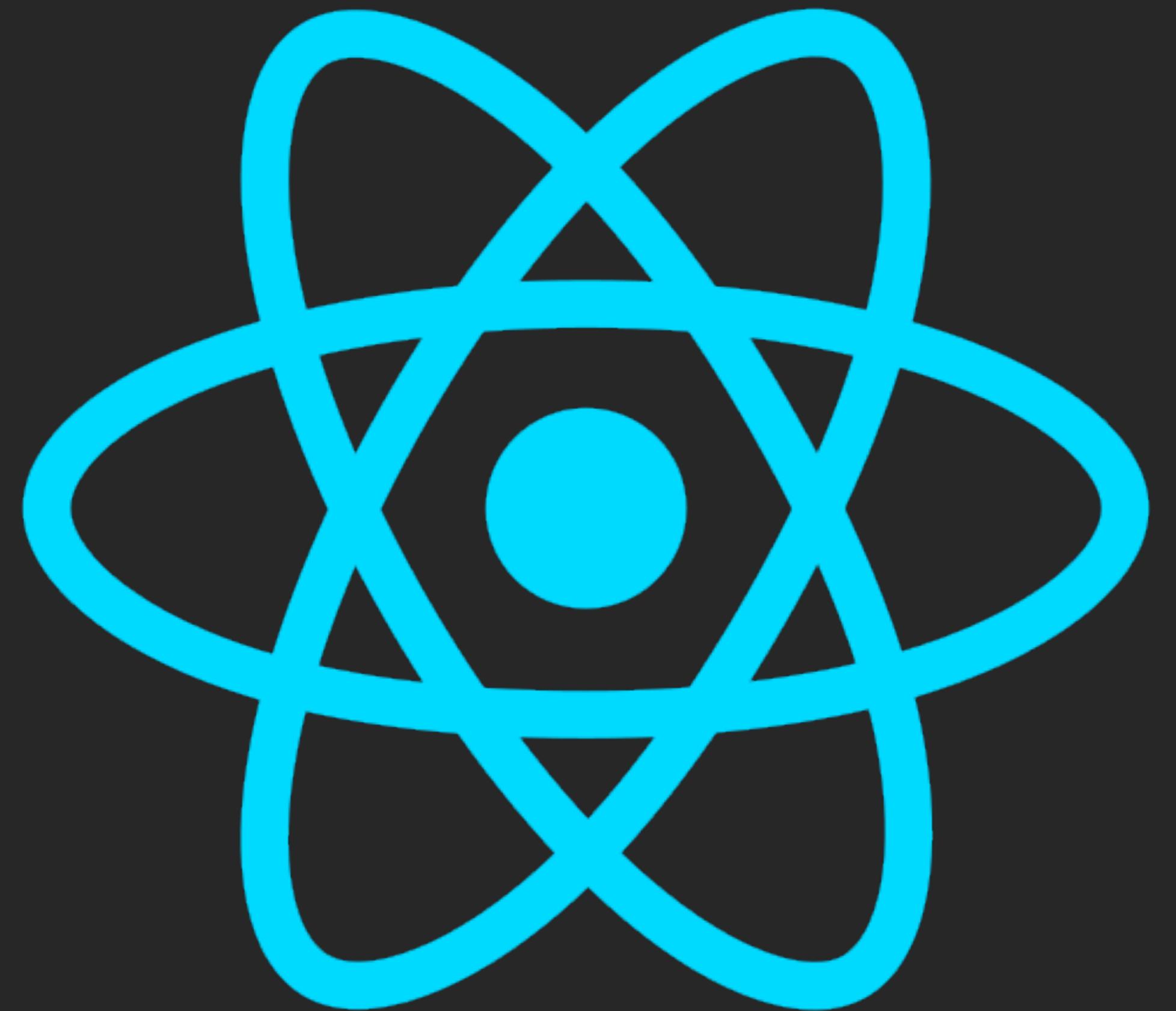


Use your front-end
flavor of choice



Deploy your app
anywhere





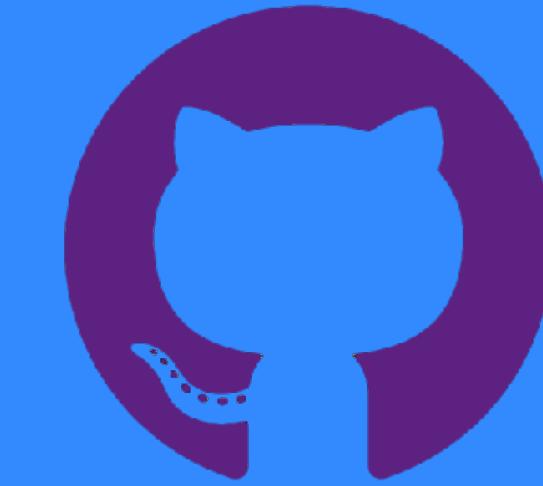
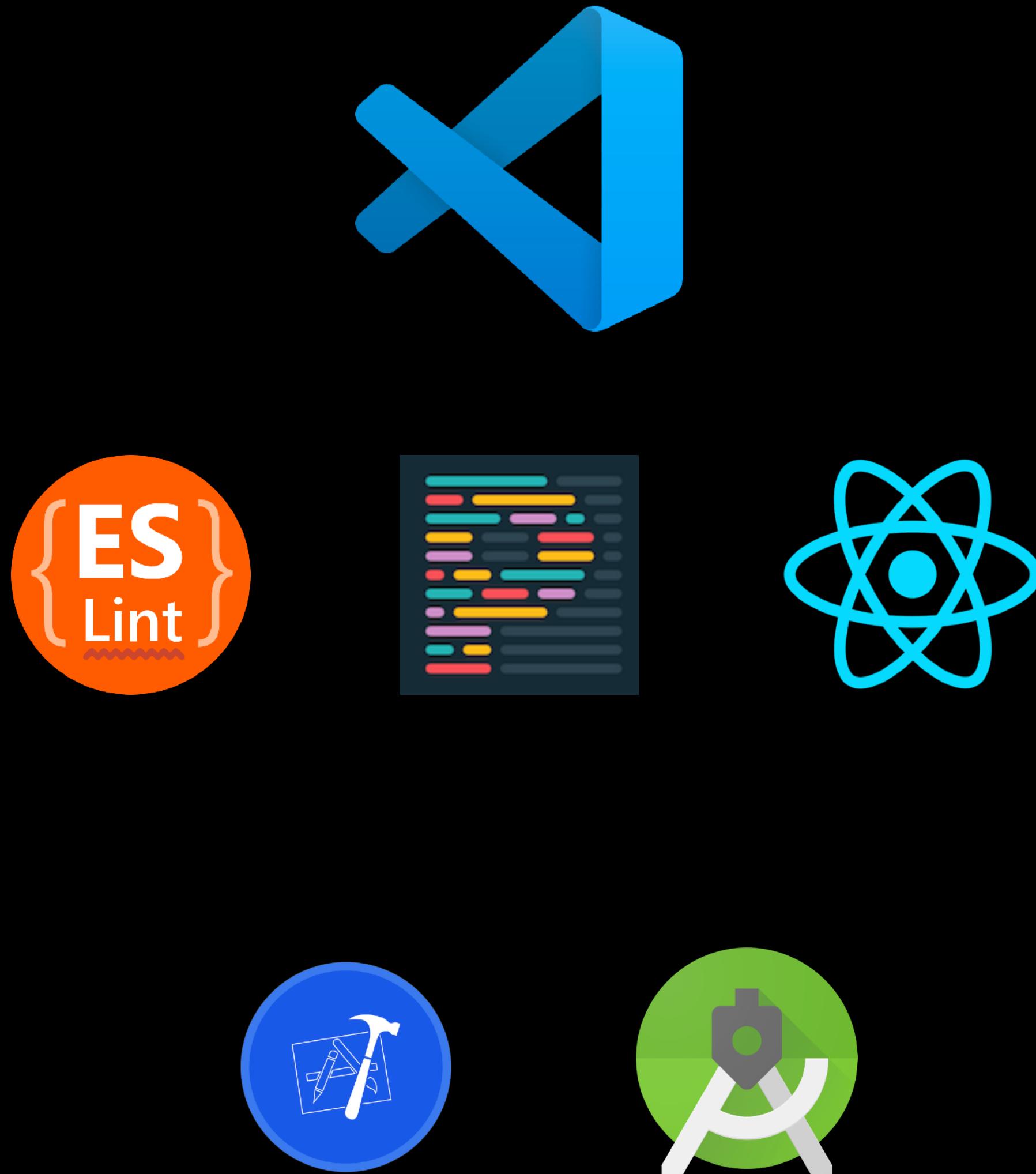
It's Just React
Ionic React is just React
components

JS

Well, at the end of
the day, it is all just
JavaScript



CLI & Tools

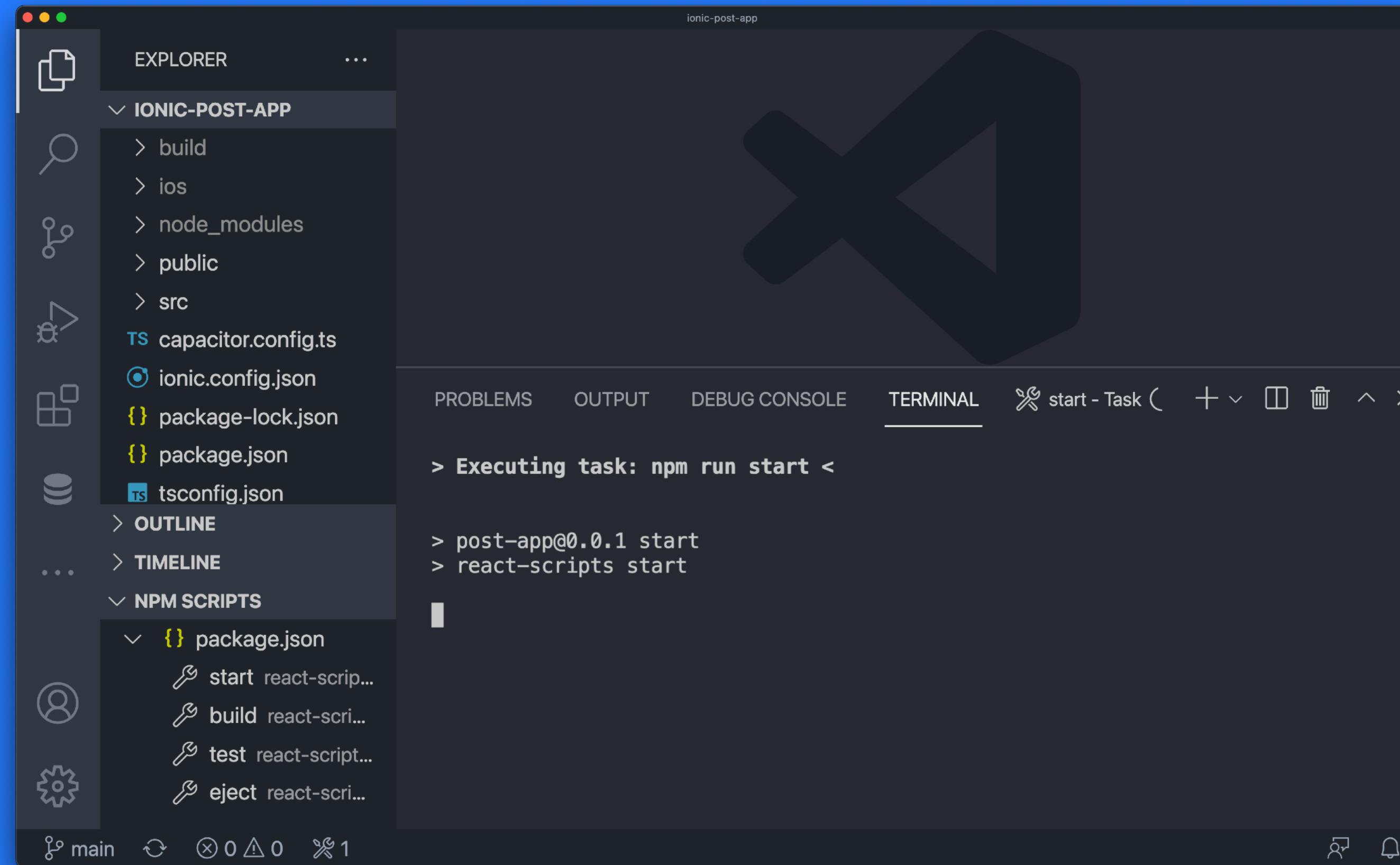


The screenshot shows a web browser window displaying the Ionic CLI Documentation. The URL in the address bar is <https://ionicframework.com/docs/cli/>. The page title is "Ionic CLI". The main content area features a large heading "Ionic CLI" followed by a paragraph: "The Ionic command-line interface (CLI) is the go-to tool for developing Ionic apps." Below this, there are two sections: "Installation" and "Help". The "Installation" section contains the command: `$ npm install -g @ionic/cli`. The "Help" section notes that the CLI ships with command documentation accessible via the `--help` flag. On the left side, there is a sidebar with navigation links for "CLI Documentation" (Overview, Configuration, Live Reload, Using a Proxy, Changelog) and "Command Reference" (build, capacitor add, capacitor build, capacitor copy, capacitor open, capacitor run, capacitor sync, capacitor update, completion, config get, config set, config unset, cordova build, cordova compile). The top navigation bar includes links for Guide, Components, CLI (which is active), Native, v6, Search, and various community and support links.

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

CLI

<https://ionicframework.com/docs/intro/cli>



```
ionic-post-app — node - npm start TERM_PROGRAM=...  
Last login: Wed Feb 2 15:07:34 on ttys003  
[race@Rasmuss-MacBook-Pro ionic-post-app % npm start  
> post-app@0.0.1 start  
> react-scripts start
```

CLI Installation

<https://ionicframework.com/docs/intro/cli>

Ionic React Quickstart

<https://ionicframework.com/docs/react/quickstart>

Getting Started

[https://race.notion.site/Getting-
Started-5086f9c2cb684f34b4cd259b4b0c44f0](https://race.notion.site/Getting-Started-5086f9c2cb684f34b4cd259b4b0c44f0)

Structure

It's all components

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

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

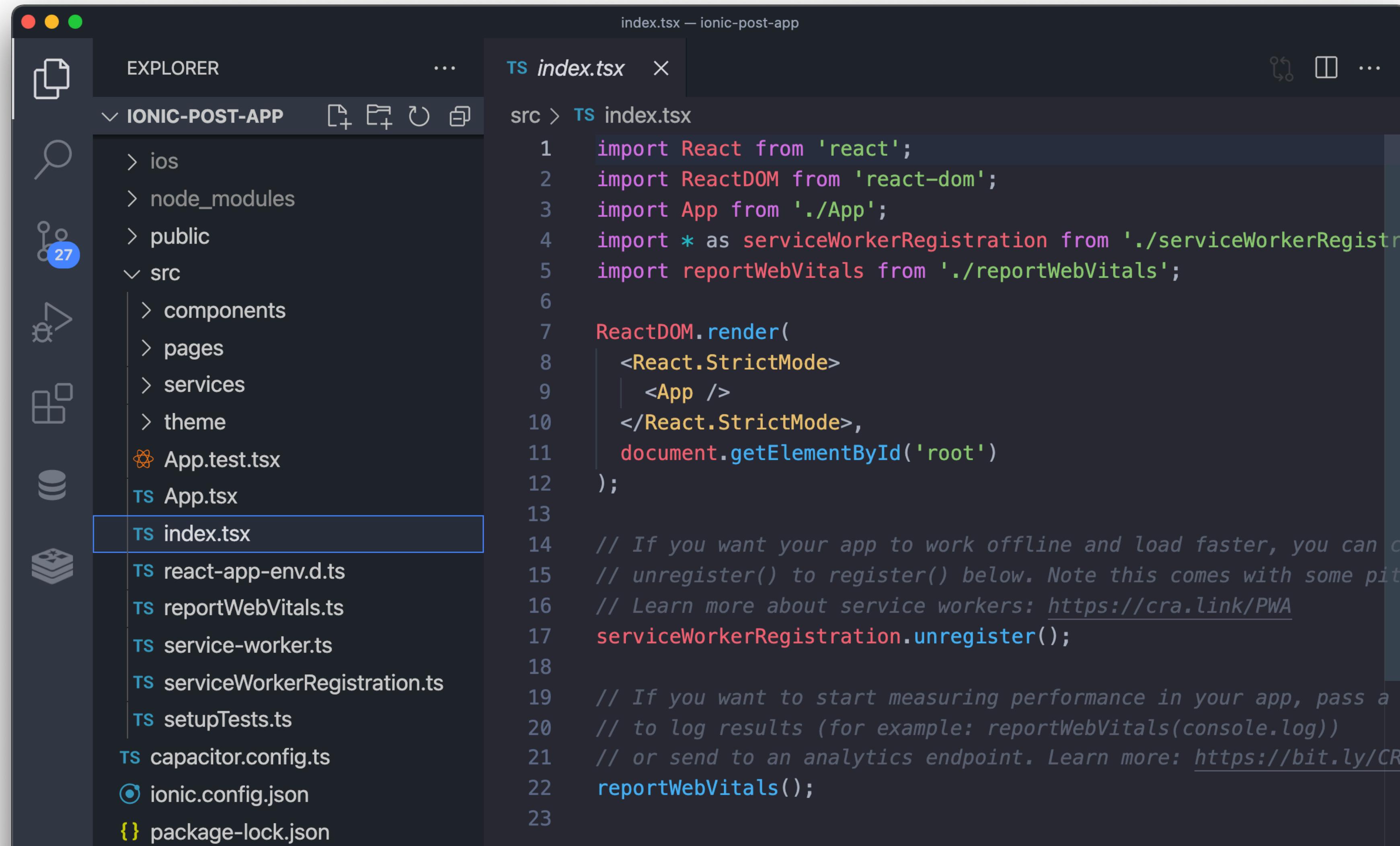
```
import React from 'react';
import { Route } from 'react-router-dom';
import { IonApp, IonRouterOutlet } from '@ionic/react';
import { IonReactRouter } from '@ionic/react-router';
import Home from './pages/Home';

/* Core CSS required for Ionic components to work properly */
import '@ionic/react/css/core.css';

const App: React.FC = () => (
  <IonApp>
    <IonReactRouter>
      <IonRouterOutlet>
        <Route path="/home" component={Home} exact={true} />
        <Route exact path="/" render={() => <Redirect to="/home" />} />
      </IonRouterOutlet>
    </IonReactRouter>
  </IonApp>
)
```

Structure

It's all components



The screenshot shows the VS Code interface with a dark theme. The Explorer sidebar on the left displays the project structure of 'IONIC-POST-APP' with a count of 27 files. The 'src' directory contains 'components', 'pages', 'services', 'theme', 'App.test.tsx', 'App.tsx', and 'index.tsx'. The 'index.tsx' file is currently selected and open in the main editor area. The code in 'index.tsx' is a standard React application entry point:

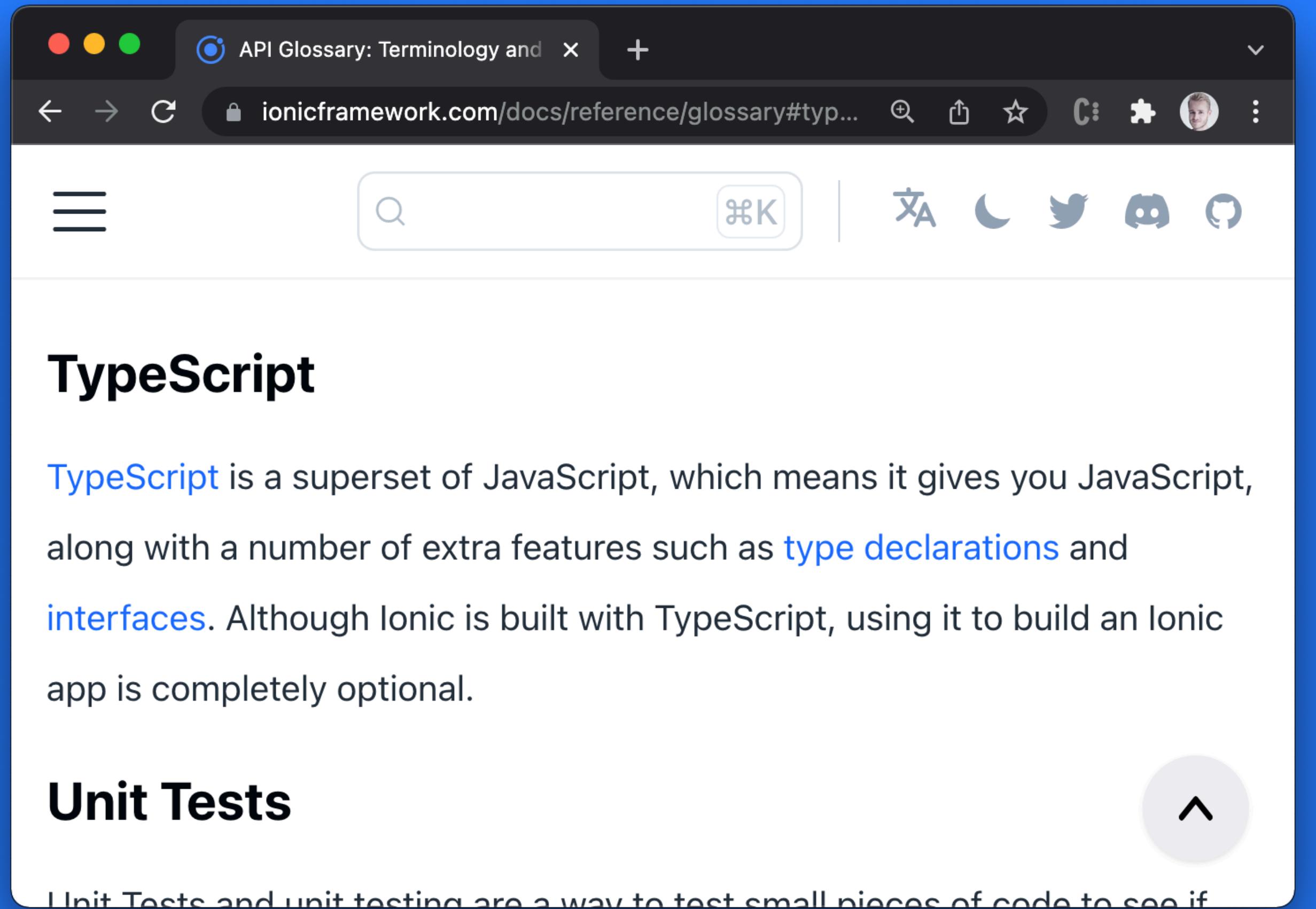
```
index.tsx — ionic-post-app
TS index.tsx  X
src > TS index.tsx
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import App from './App';
4 import * as serviceWorkerRegistration from './serviceWorkerRegistration';
5 import reportWebVitals from './reportWebVitals';

6
7 ReactDOM.render(
8   <React.StrictMode>
9     | <App />
10    </React.StrictMode>,
11    document.getElementById('root')
12  );
13
14 // If you want your app to work offline and load faster, you can
15 // unregister() to register() below. Note this comes with some
16 // performance overhead so don't use in production.
17 // Learn more about service workers: https://cra.link/PWA
18 serviceWorkerRegistration.unregister();

19 // If you want to start measuring performance in your app, pass a
20 // to log results (for example: reportWebVitals(console.log))
21 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-alytics
22 reportWebVitals();
23
```

.tsx?

What the ... TypeScript 



A screenshot of a web browser window. The title bar says "API Glossary: Terminology and". The address bar shows "ionicframework.com/docs/reference/glossary#glossary#typ...". Below the address bar is a header with a search icon, a logo, and social media links for GitHub, Twitter, and others. The main content area has a section titled "TypeScript" with the following text:

TypeScript is a superset of JavaScript, which means it gives you JavaScript, along with a number of extra features such as type declarations and interfaces. Although Ionic is built with TypeScript, using it to build an Ionic app is completely optional.

Below this is a section titled "Unit Tests" with a small explanatory text and a circular arrow icon.

<https://ionicframework.com/docs/reference/glossary#glossary#typescript>

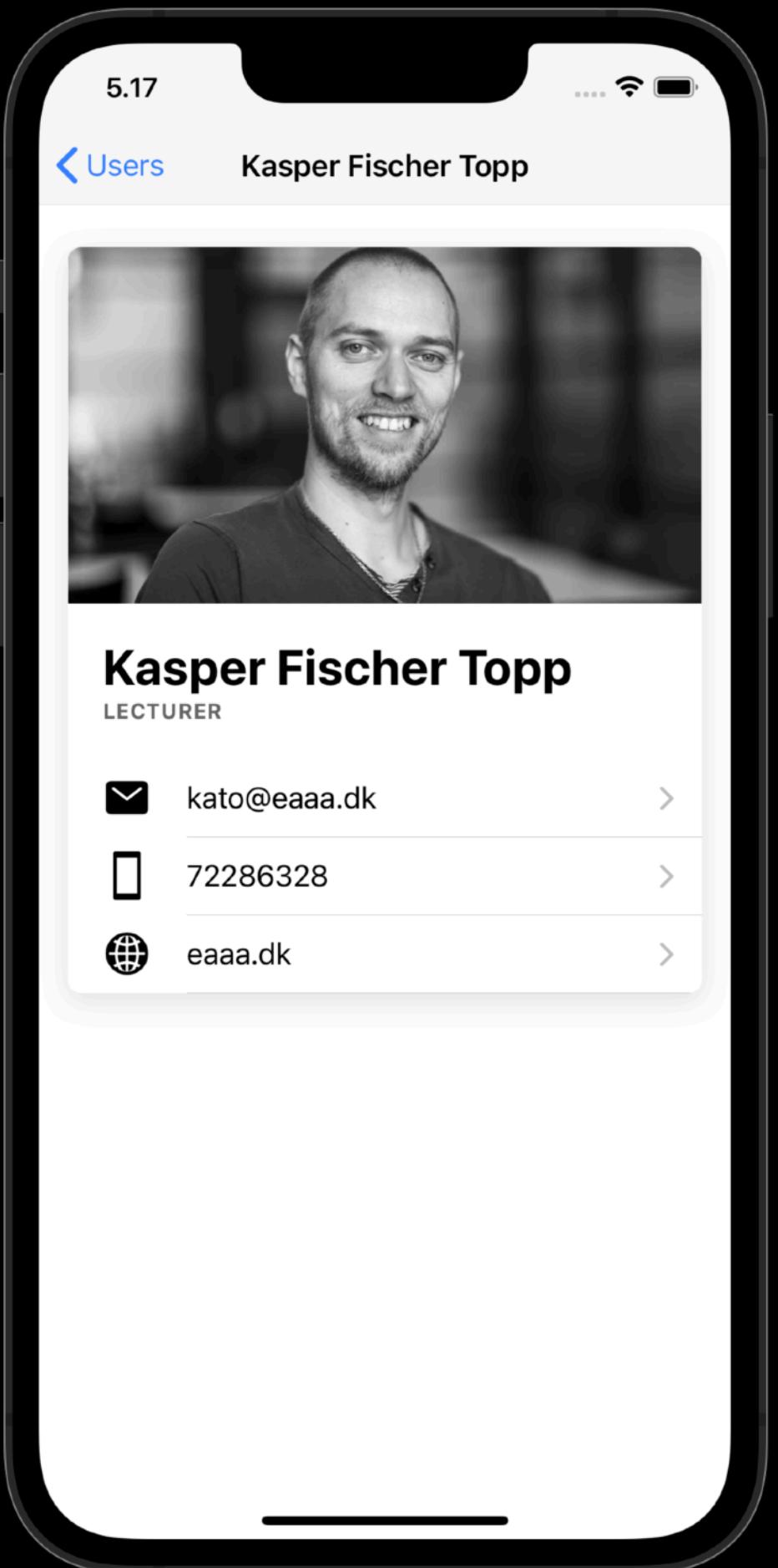
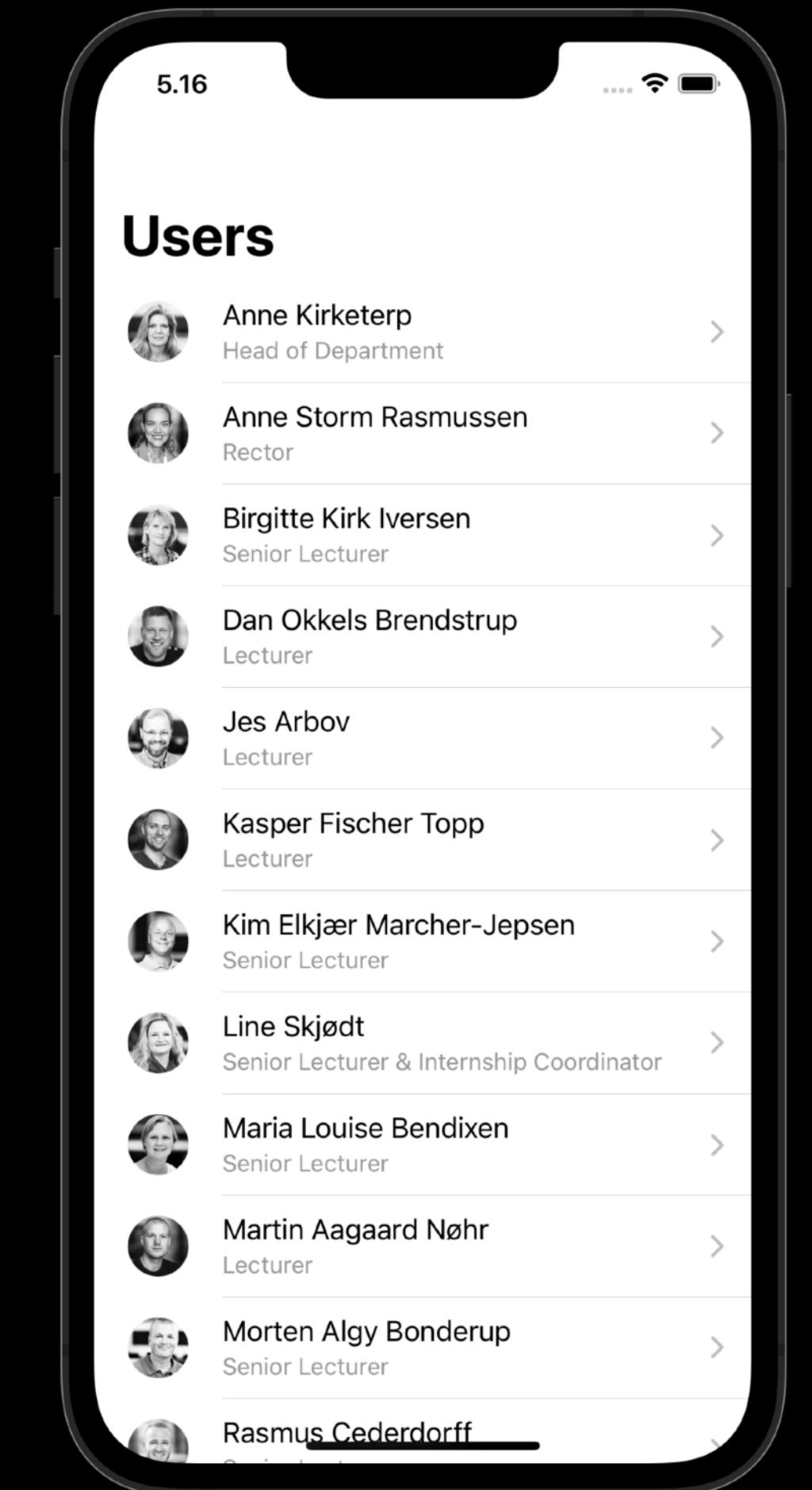
TypeScript is **JavaScript with syntax for types**.

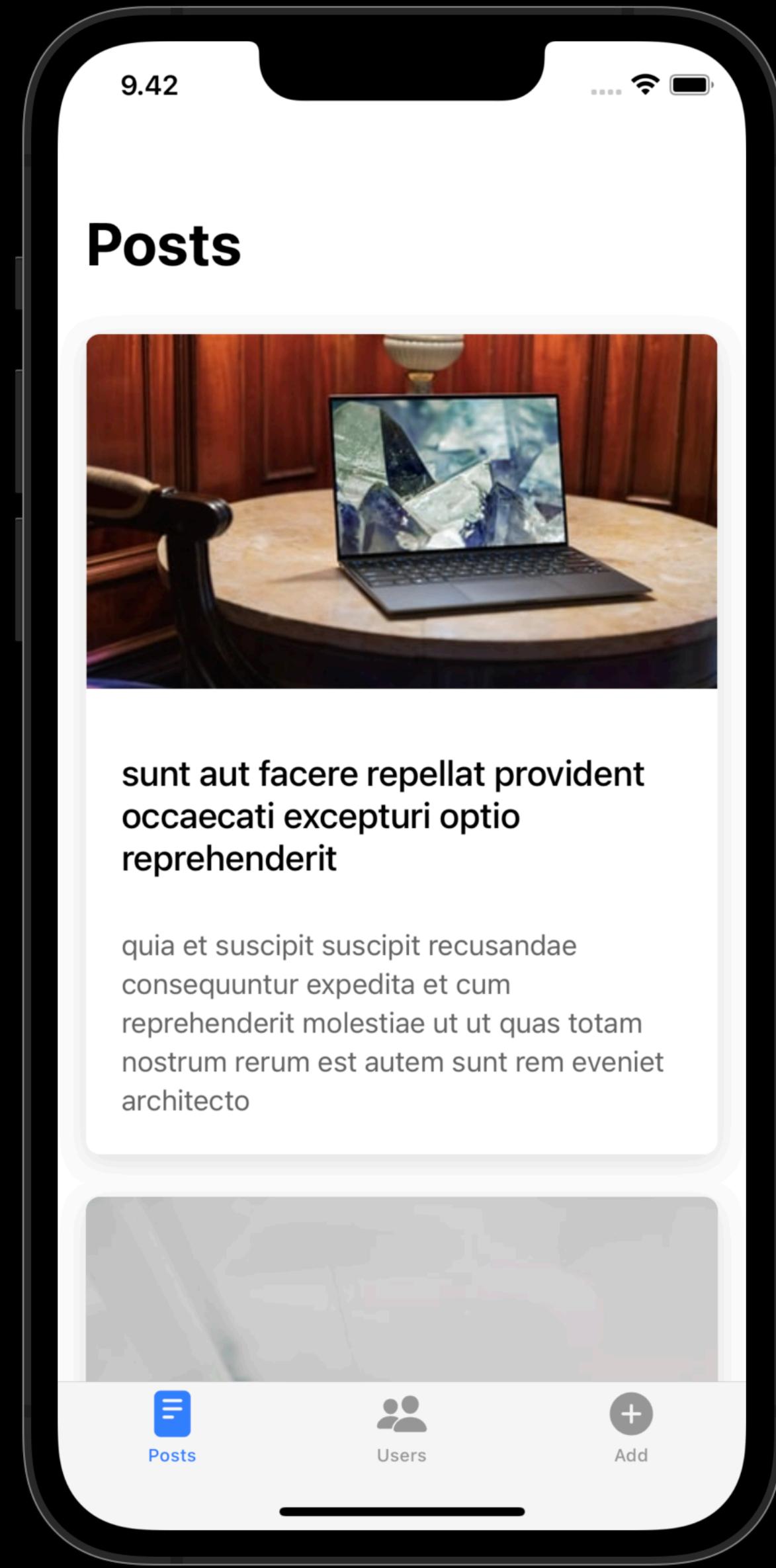
TypeScript is a strongly typed programming language that builds on JavaScript, giving you better tooling at any scale.

<https://www.typescriptlang.org/>

Ionic User List App

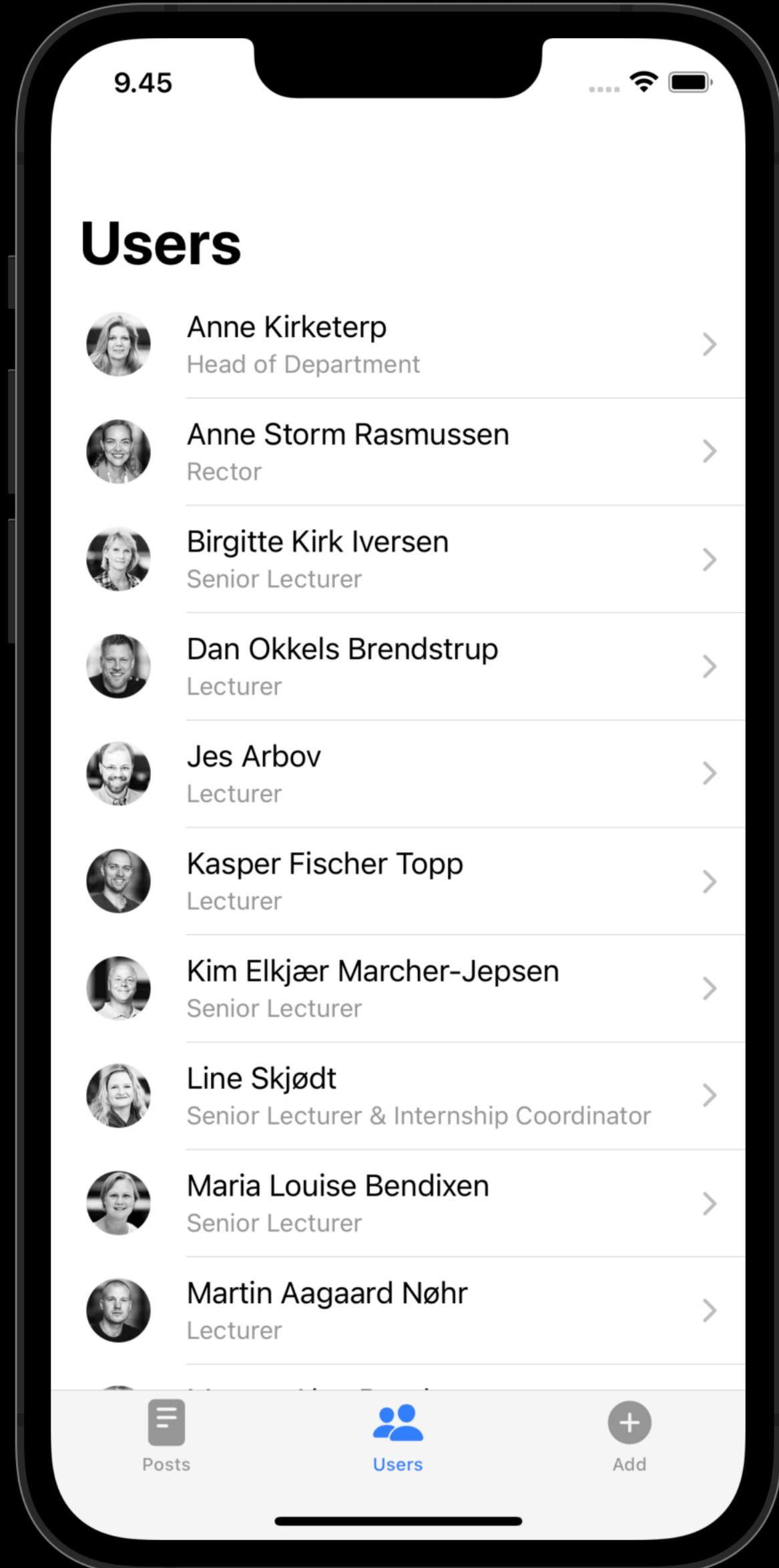
<https://race.notion.site/User-List-e4d5af63b2de443f946cc2f5a1227716>





Ionic Post App

1. Use tabs starter template.
2. Customise the template with Posts Page, Users Page and Add Page.
3. In the Posts tab, fetch and display posts from this source.
4. Display image, title and body property using Ionic UI Components.



Ionic Post App

5. Add the list of users in the second tab, Users Page.
6. Make sure you can navigate to the detail view.

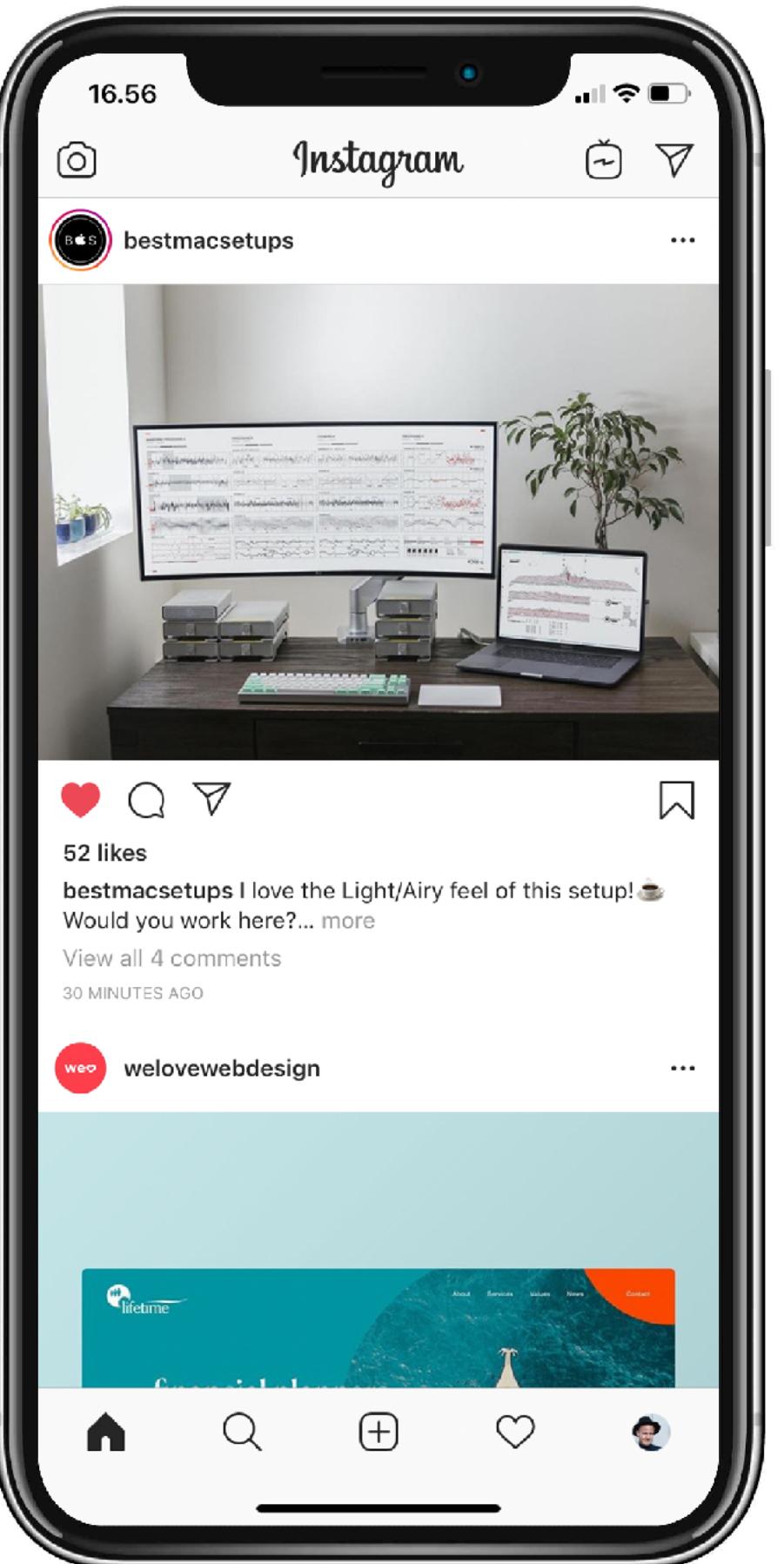
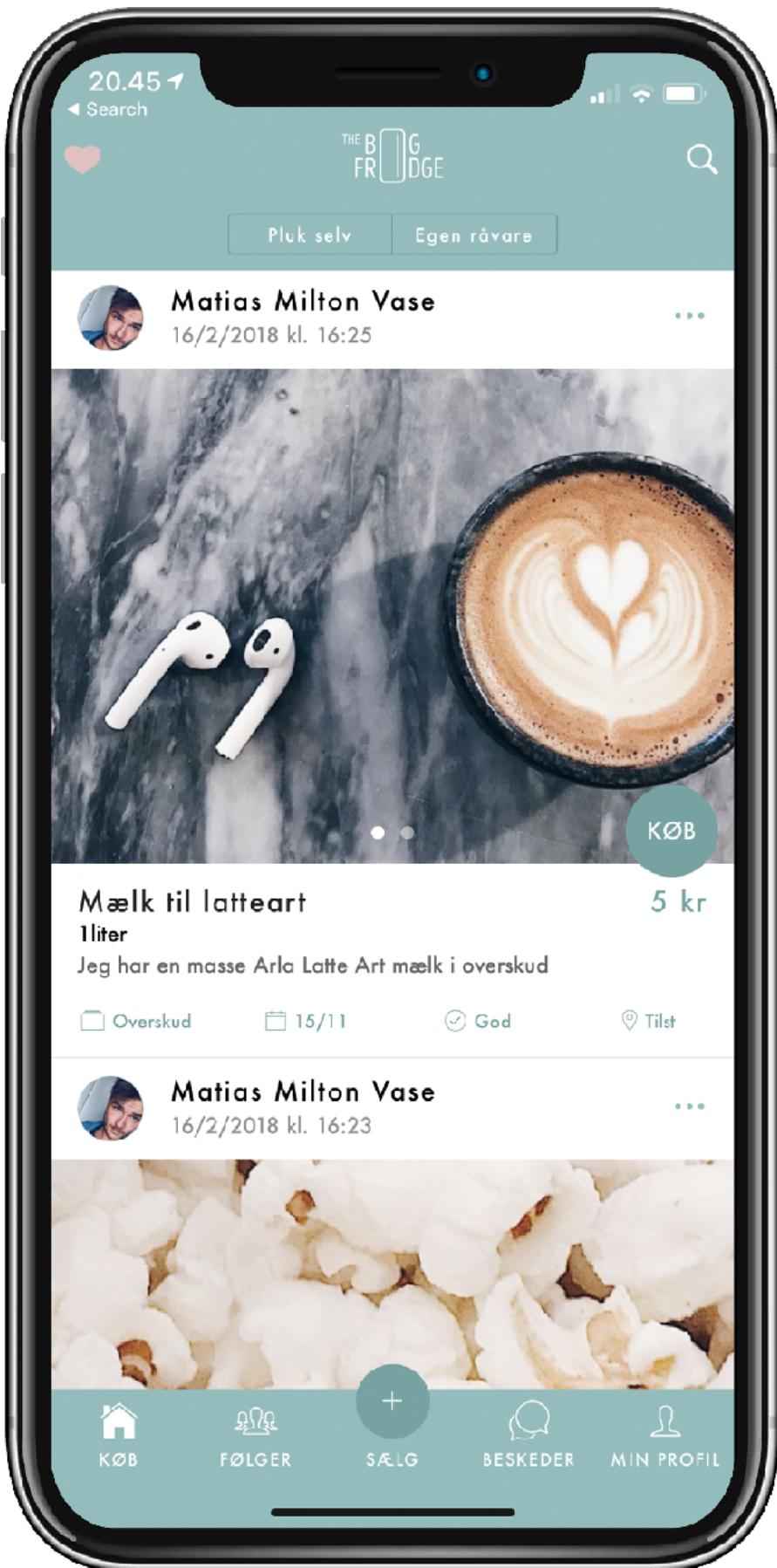
Ionic Post App

7. Think in components. Is there any logic that might be useful to extract to separate components?
8. Reimplement your pages using components.

```
    <div>
      <ul>
        <li><img alt="checkmark icon" /> src
          <ul>
            <li><img alt="checkmark icon" /> components
              <ul>
                <li>TS PostListItem.tsx</li>
                <li>TS UserListItem.tsx</li>
              </ul>
            </li>
            <li><img alt="checkmark icon" /> pages
              <ul>
                <li>TS Add.tsx</li>
                <li>TS Posts.tsx</li>
                <li>TS User.tsx</li>
                <li>TS Users.tsx</li>
              </ul>
            </li>
            <li>> services</li>
            <li>> theme</li>
            <li><img alt="cross icon" /> App.test.tsx</li>
            <li>TS App.tsx</li>
            <li>TS index.tsx</li>
          </ul>
        </li>
      </ul>
    </div>
```

Display User

... on given post (uid)

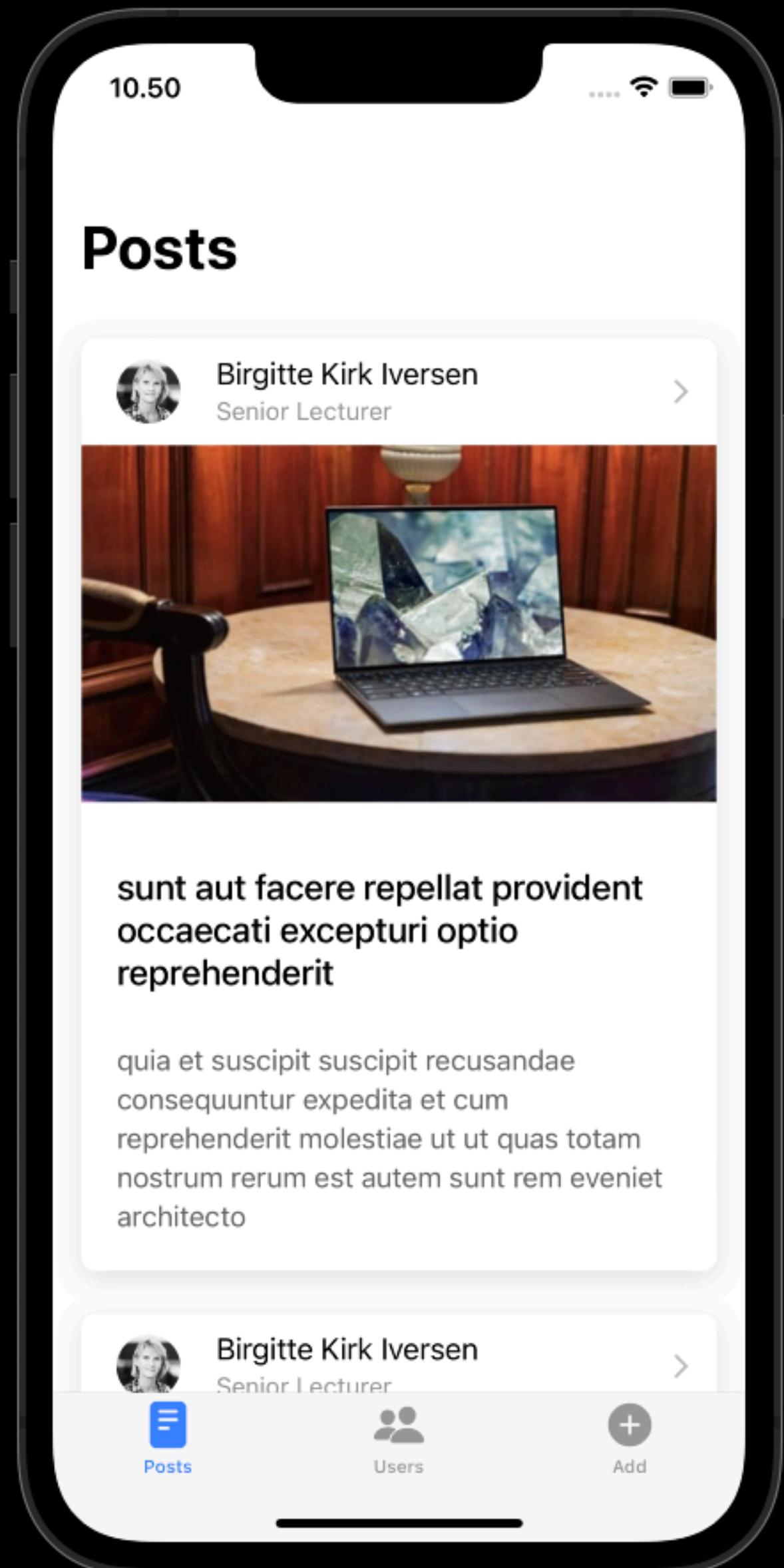


Ionic Post App

9. Improve your post view by adding user details (post create by...). Start by adding hardcoded user details like user avatar and user name using Ionic UI Components.

10. Implement logic to concatenate the posts array with user data from users array. All post objects have a `uid` property.

11. Use the concatenated array to display posts with user details dynamically.



Combine posts with users

```
import userService from "./usersService";

class PostService {
  >   constructor() { ... }

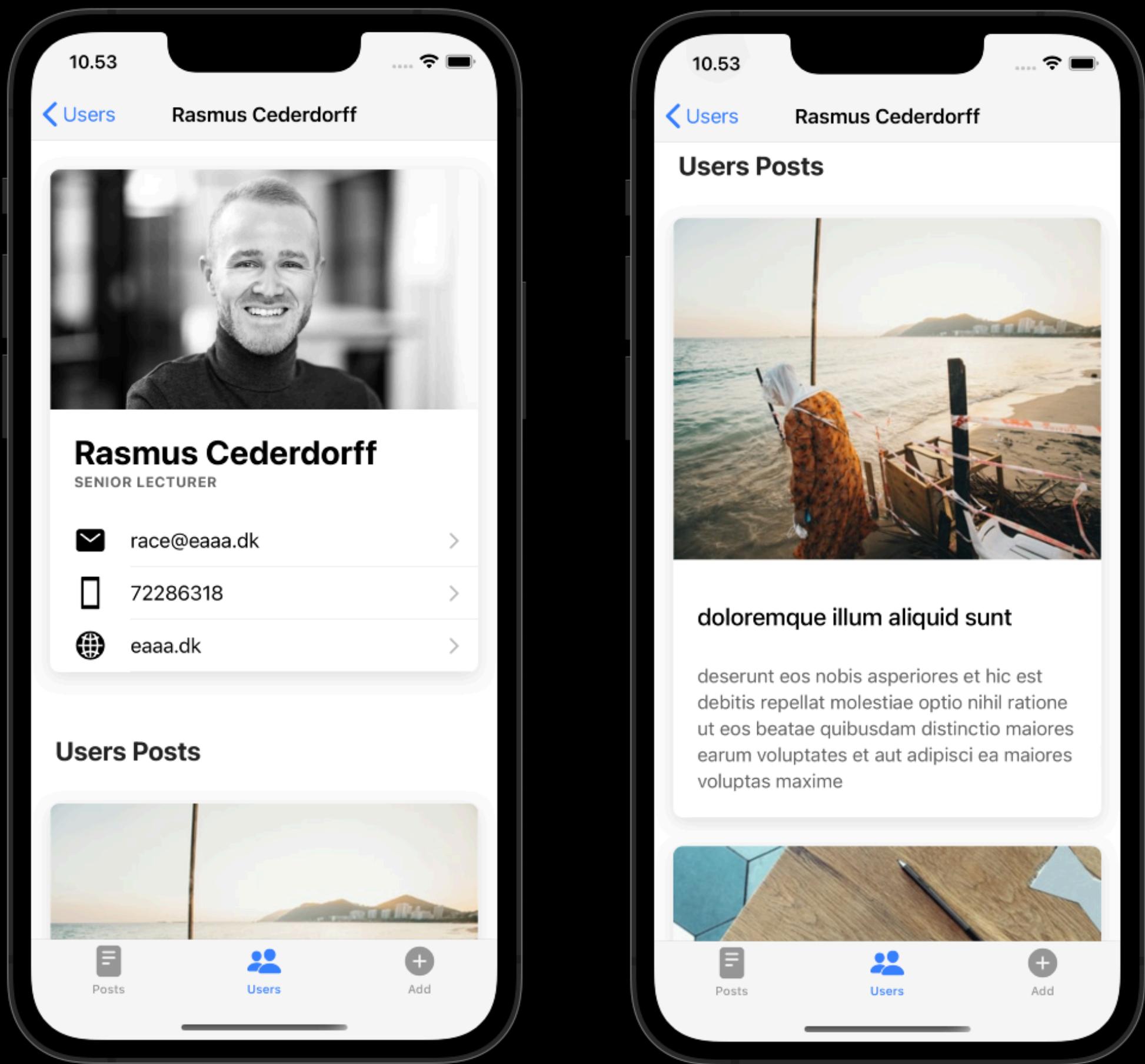
  >   async fetchPosts() { ... }

  >   async getPosts() { ... }

  async getPostsWithUserDetails() {
    if (this.posts.length === 0) {
      await this.fetchPosts();
    }
    const users = await userService.getUsers();

    const postsWithUser = this.posts.map(post => {
      const user = users.find(user => user.id === post.uid);
      post = { ...post, user: user }; // combine objects with spread operator
      delete post.uid; // delete uid - it's inside post.user.id
      return post;
    });
    return postsWithUser;
  }
}
```

Ionic Post App



12. Improve the user detail view by displaying all posts created by the user. Again, think of the `uid` property on every post object and filter posts by given `uid`.

13. Components, components, components. When implementing improvements, think of how to (re-)use your existing components. No code duplicating is allowed.

Ionic Post App

14. Implement the missing tab, Add.

15. Explore Ionic UI Components.

16. Implement the Add tab with UI Components to create a new post object with the following properties: uid, id, title, body & image.

uid can be hardcoded. image can be a URL.

17. The new post object must be pushed to the array of posts.



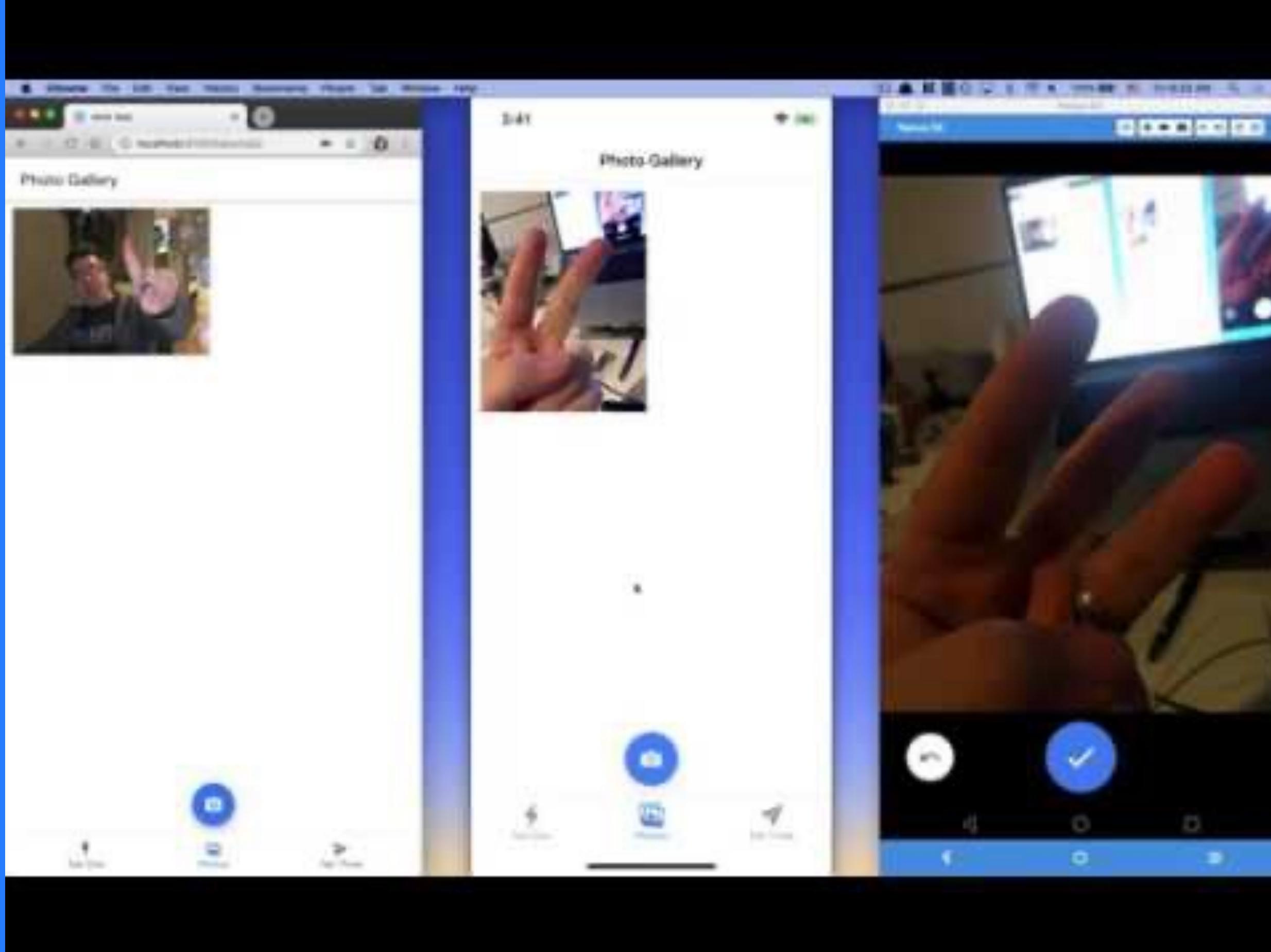
Components

Think in components. Is there any logic that might be useful to move to separate components?

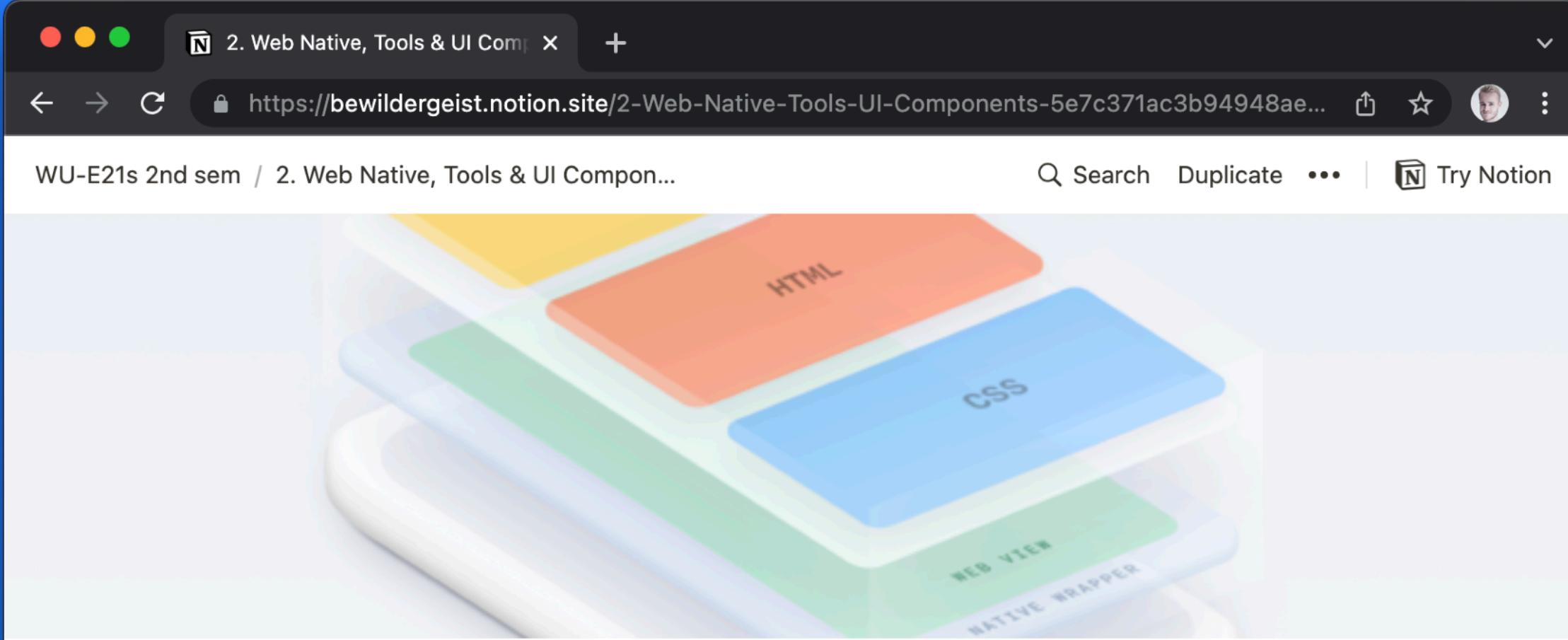
```
✓ src
  ✓ components
    TS PostListItem.tsx
    TS UserListItem.tsx
  ✓ pages
    TS Add.tsx
    TS Posts.tsx
    TS User.tsx
    TS Users.tsx
  > services
  > theme
  ⚡ App.test.tsx
  TS App.tsx
  TS index.tsx
```

Ionic React Camera App

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



Next Tuesday



The screenshot shows a Notion page with the following details:

- Title:** 2. Web Native, Tools & UI Components
- Date:** 10/02/2022
- Teacher:** RACE
- Course:** MAD

Agenda/ Themes

- Ionic React & typescript
- Ionic Core Concepts
- UI Components
- Mobile UI & UX
- Page Navigation & IonTabs
- Routing, React Router & IonReactRouter