WEB NATIVE, TOOLS & UI COMPONENTS

Agenda

Ionic React & TypeScript

Ionic Core Concepts

UI Components

Mobile Navigation & Routing

(Mobile UI & UX)

Your First Ionic Apps

Ionic User List App

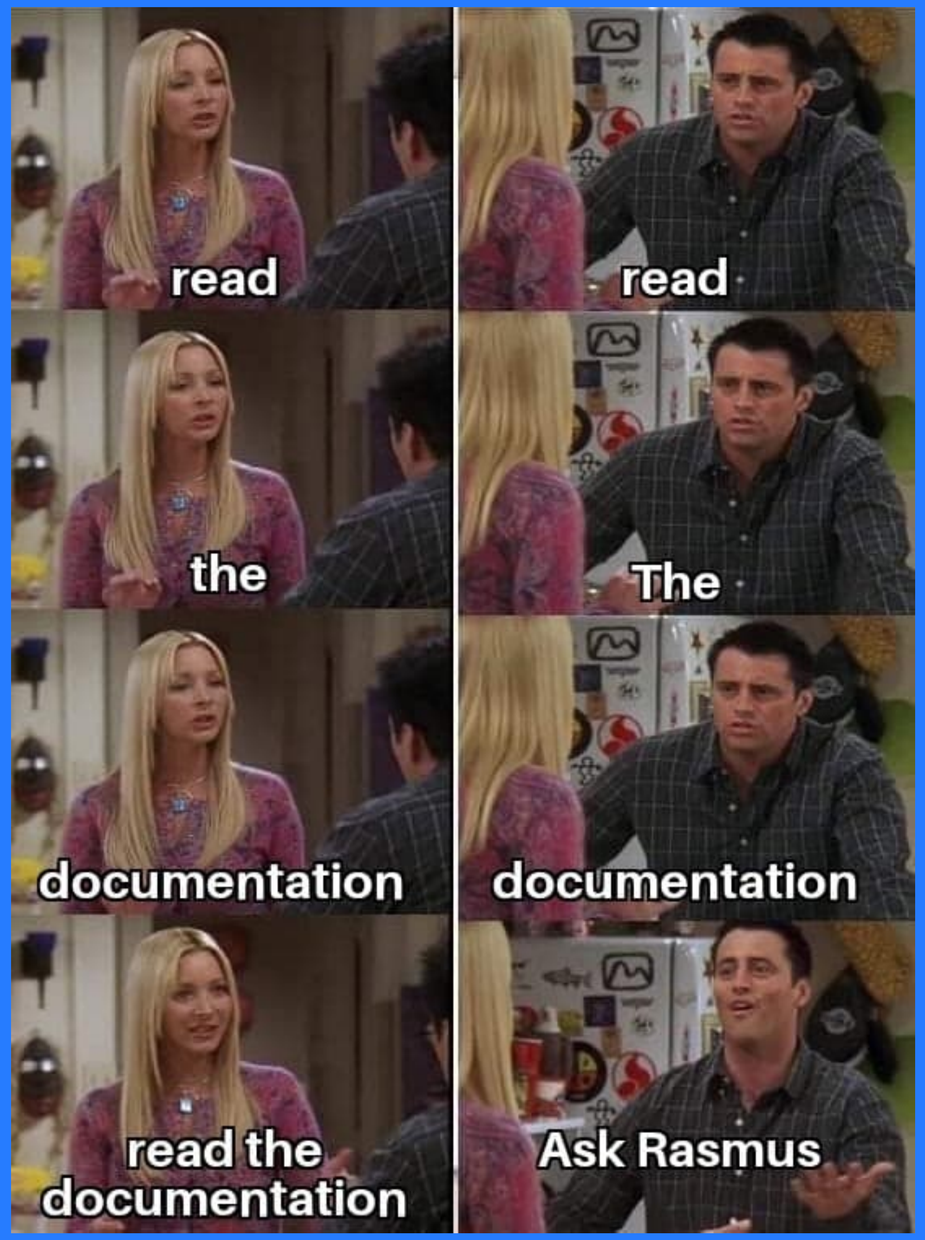
Thinking in React?

Components, components, components

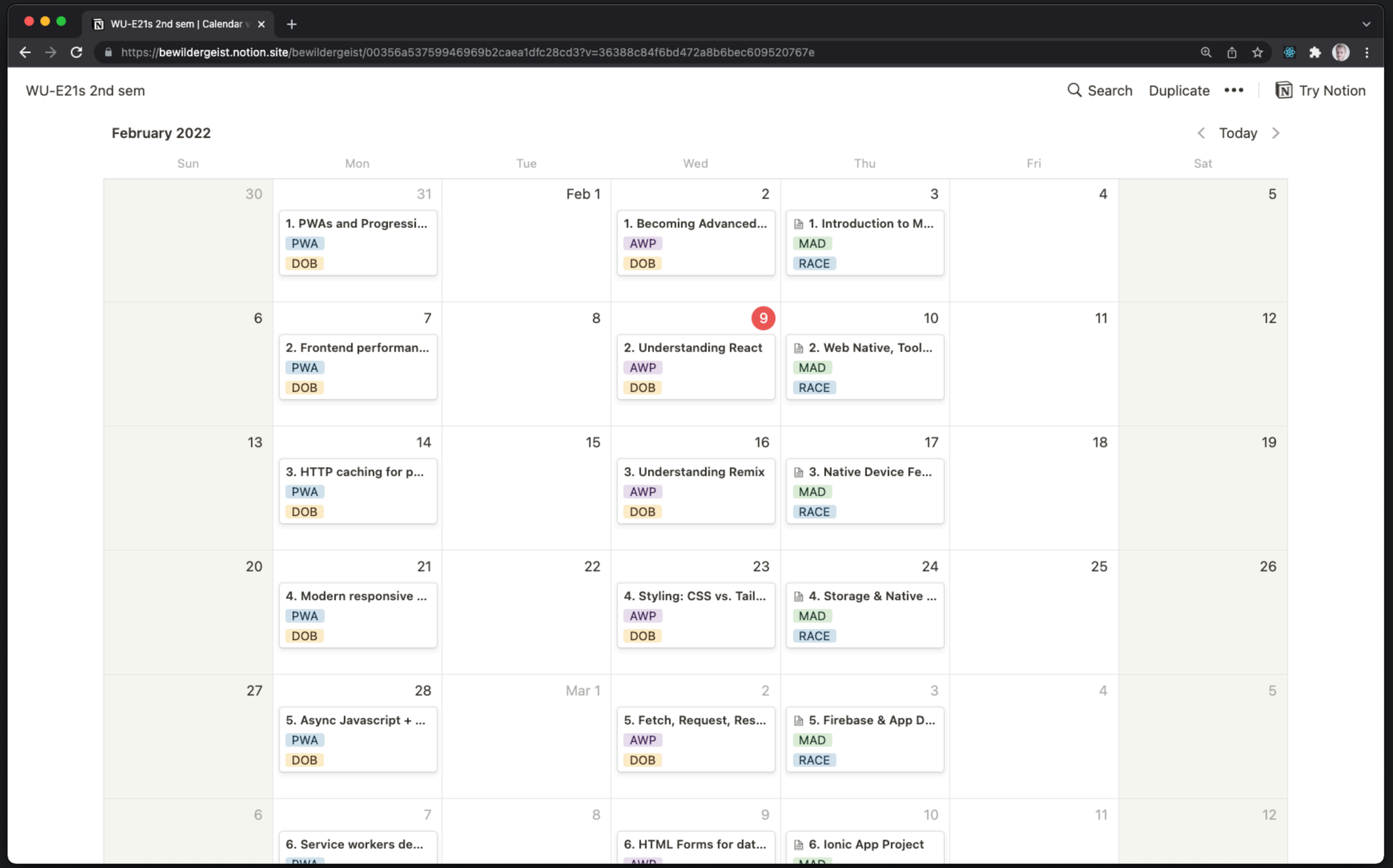
Why lifecycle methods?

Ionic Post App

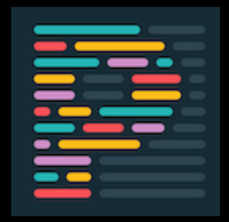
Ionic Camera App

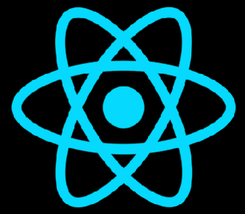


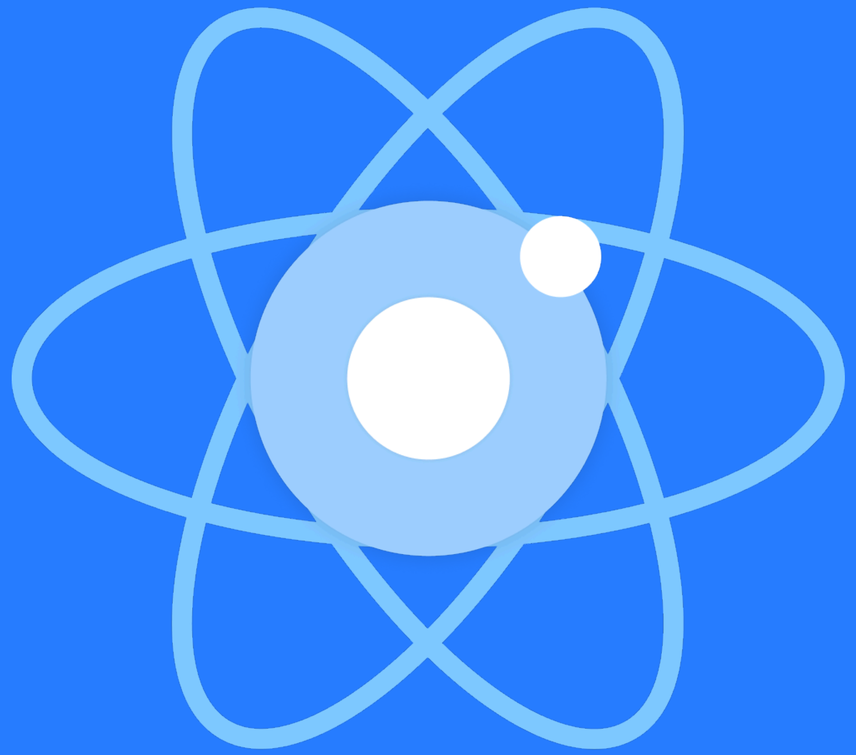
Hands-on & Documentation


WU-E21s 2nd sem | Calendar
https://bewildergeist.notion.site/bewildergeist/00356a53759946969b2caea1dfc28cd3?v=36388c84f6bd472a8b6bec609520767e
WU-E21s 2nd sem
Search
Duplicate
Try Notion
February 2022
Today
Sun
Mon
Tue
Wed
Thu
Fri
Sat
30
31
1. PWAs and Progressi...
PWA
DOB
Feb 1
2
1. Becoming Advanced...
AWP
DOB
3
1. Introduction to M...
MAD
RACE
4
5
6
7
2. Frontend performan...
PWA
DOB
8
9
2. Understanding React
AWP
DOB
10
2. Web Native, Tool...
MAD
RACE
11
12
13
14
3. HTTP caching for p...
PWA
DOB
15
16
3. Understanding Remix
AWP
DOB
17
3. Native Device Fe...
MAD
RACE
18
19
20
21
4. Modern responsive ...
PWA
DOB
22
23
4. Styling: CSS vs. Tail...
AWP
DOB
24
4. Storage & Native ...
MAD
RACE
25
26
27
28
5. Async Javascript + ...
PWA
DOB
Mar 1
2
5. Fetch, Request, Res...
AWP
DOB
3
5. Firebase & App D...
MAD
RACE
4
5
6
7
6. Service workers de...
8
9
6. HTML Forms for dat...
10
6. Ionic App Project
11
12

CLI & Tools





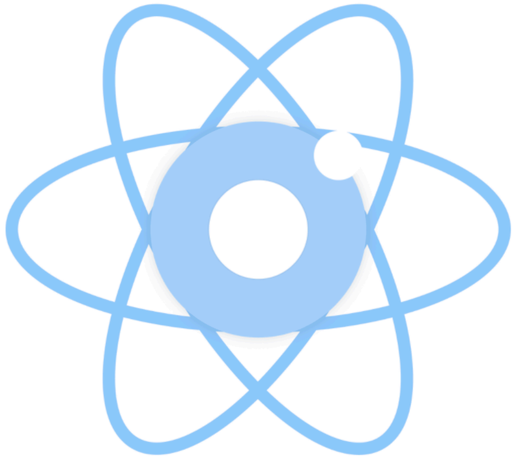


Ionic React

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

Ionic is a Cross-platform

One Codebase
Any Platform
Just React



✓ 100+ mobile optimized React UI components

✓ Standard React tooling with react-dom

✓ iOS / Android / Electron / PWA

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.

UI Components

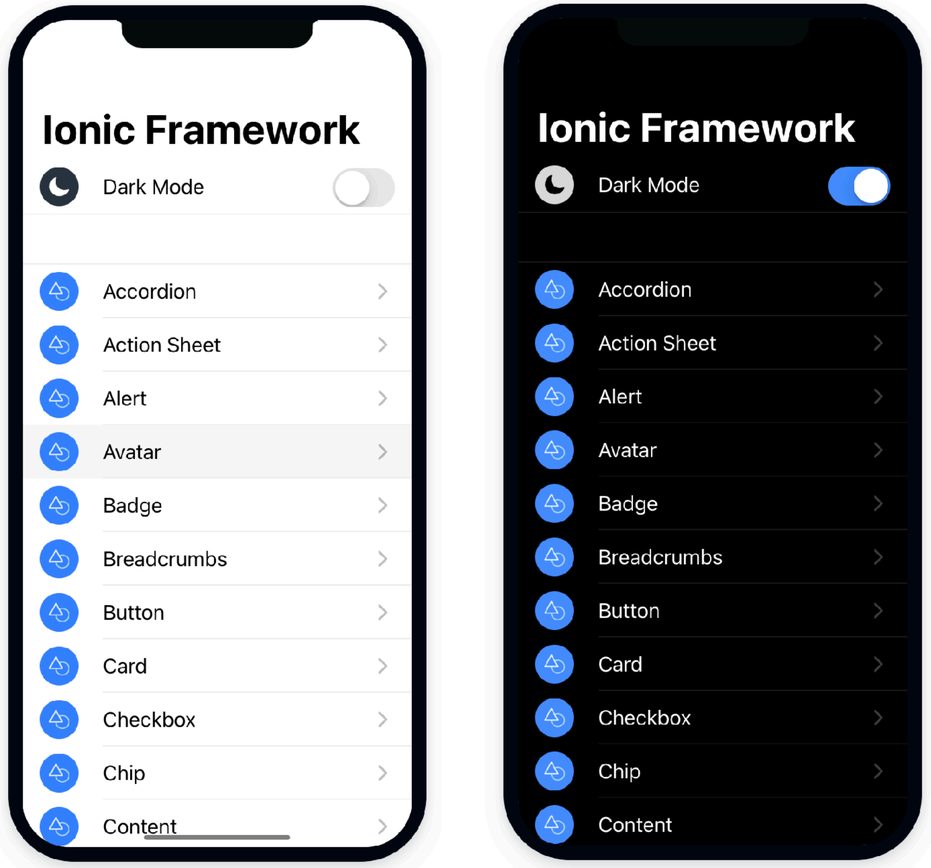
Dive into Ionic beautifully designed UI component library.

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.



https://ionicframework.com/

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.”

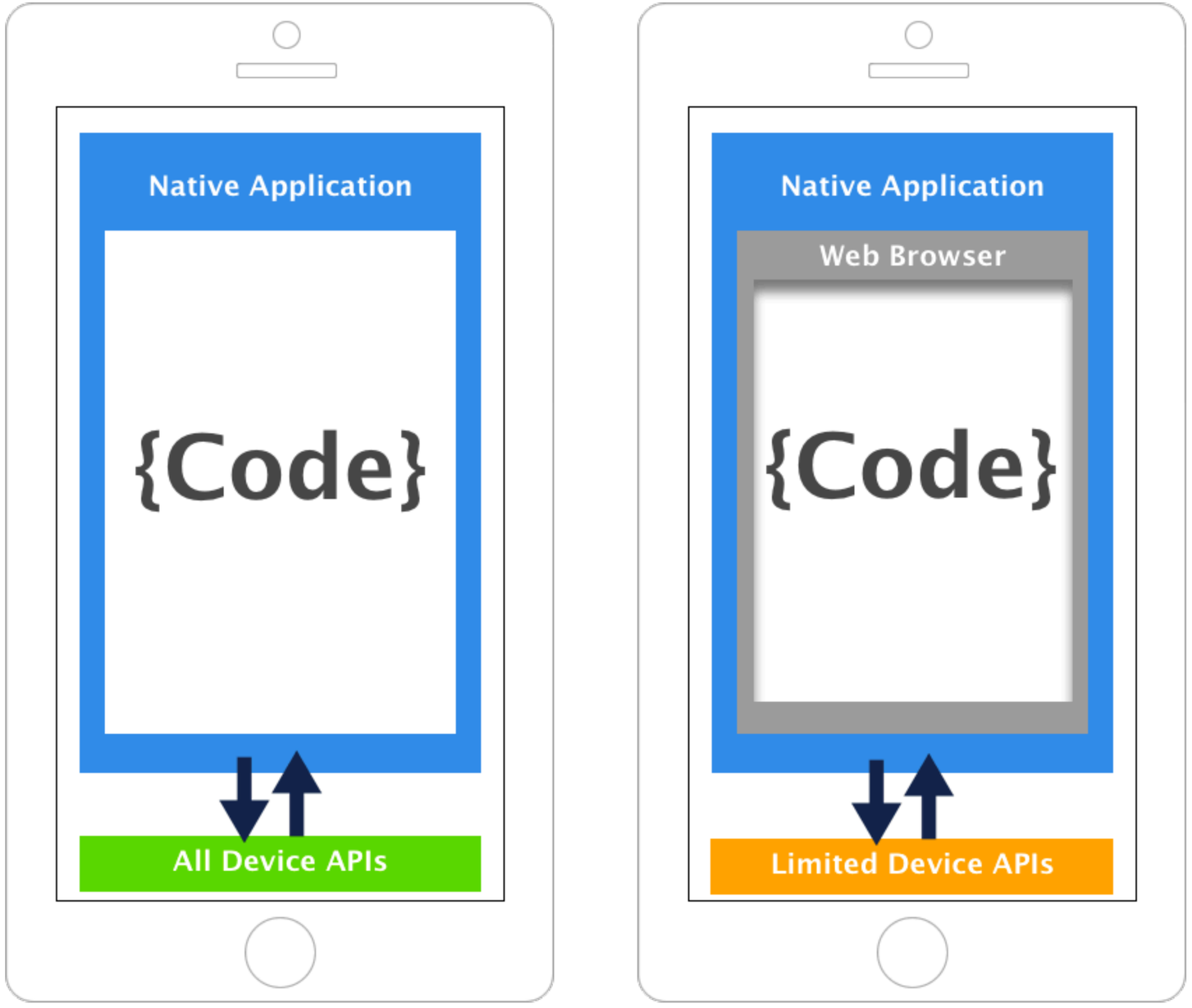
“Web Native is "hybrid" done right, and it's the future of mobile app development.”

https://webnative.tech/

Capacitor

A cross-platform native runtime for web apps.

https://capacitorjs.com/


Native Application
{Code}
All Device APIs
Native Application
Web Browser
{Code}
Limited Device APIs

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)

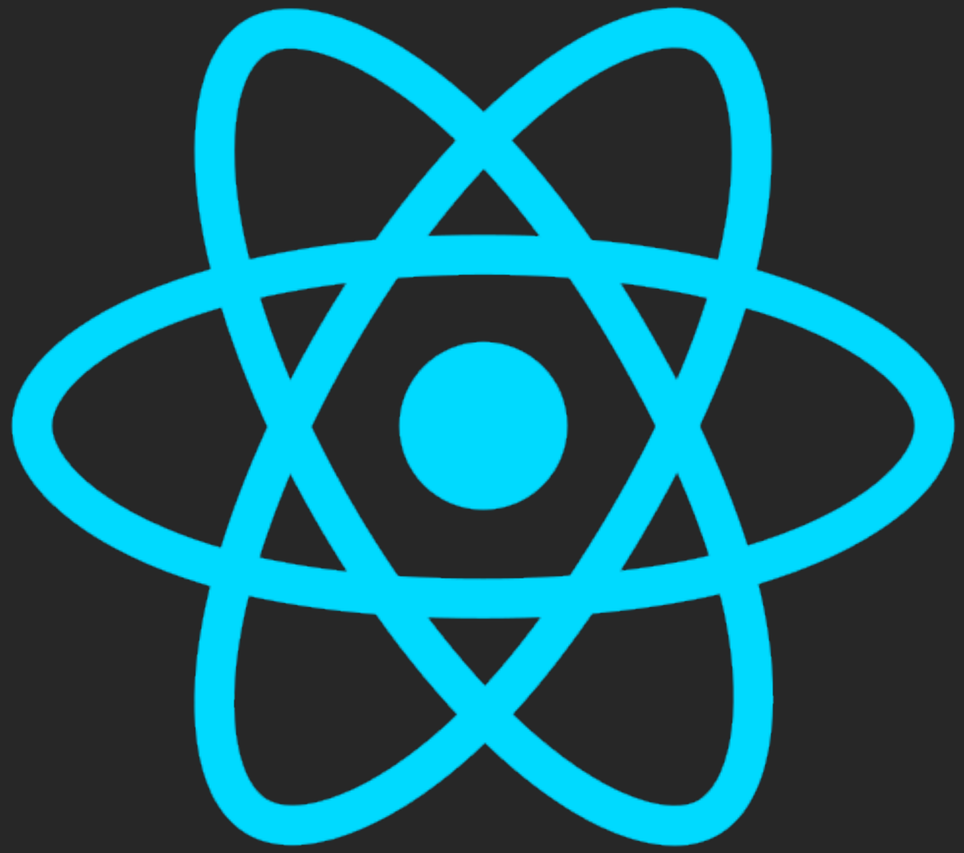
Feature	Native	Web-only	Hybrid
Device Access	Full	Limited	Full (with plugins)
Performance	High	Medium to High	Medium to High
Development Language	Platform Specific	HTML, CSS, Javascript	HTML, CSS, Javascript
Cross-Platform Support	No	Yes	Yes
User Experience	High	Medium to High	Medium to High
Code Reuse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

Runs mordern web apps natively on iOS, Android, Electron and Webmultiple platforms.

Web Navtive Apps & PWA

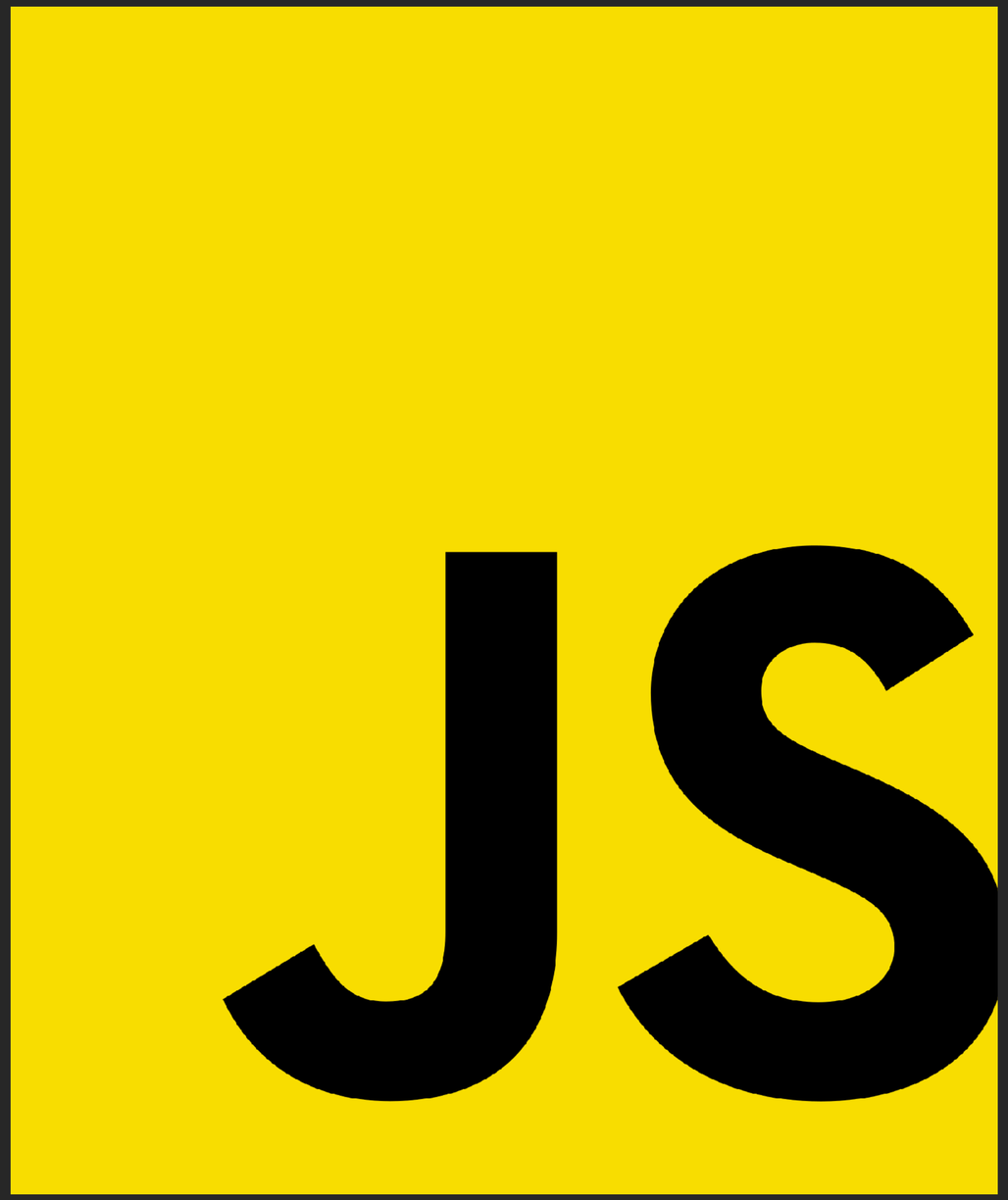
Traditional Approach

Ionic (Web Native) Approach



It’s Just React

Ionic React is just React components

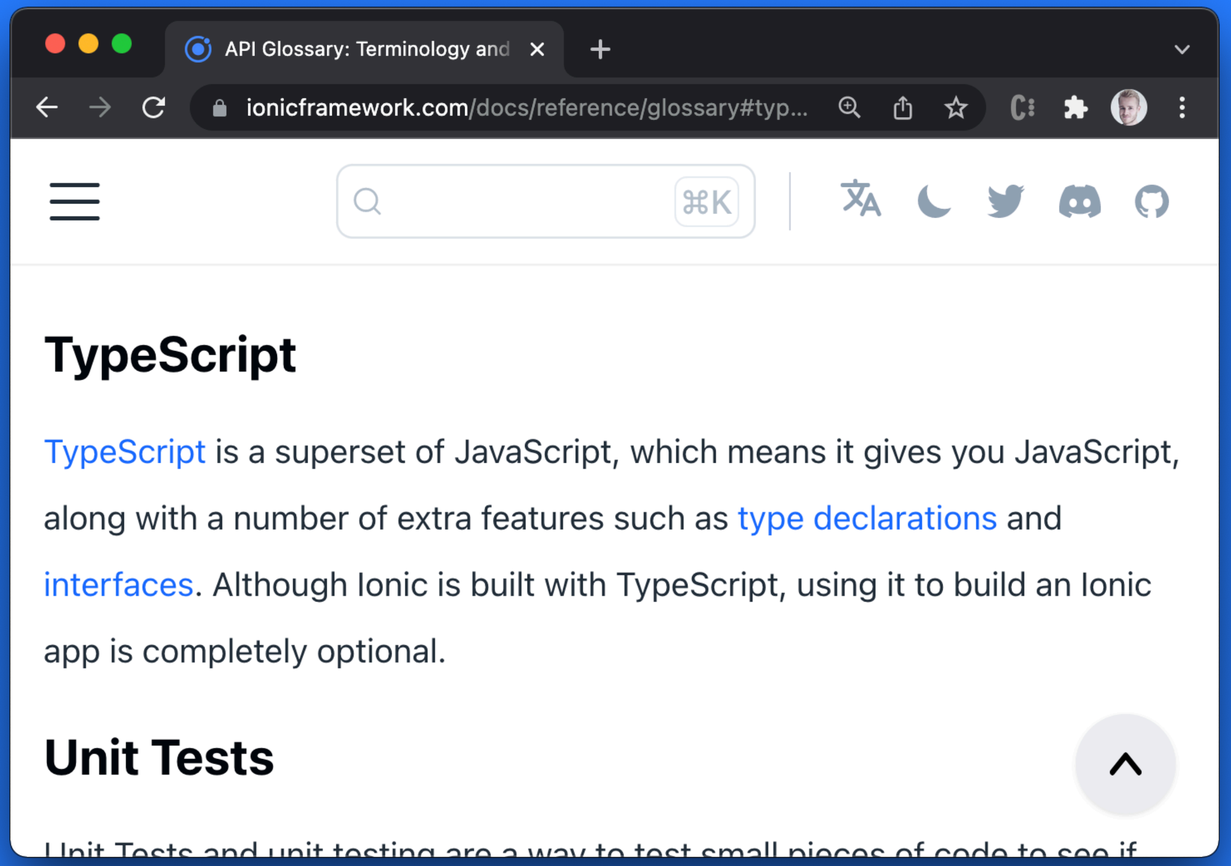


Well, at the end of the day, it’s just JavaScript

🤷🏼‍♂️

simplilearn

WHAT IS TYPESCRIPT?



https://ionicframework.com/docs/reference/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 DOCS

Action Sheet
ion-action-sheet

Accordion
ion-accordion
ion-accordion-group

Alert
ion-alert

Badge
ion-badge

Breadcrumb
ion-breadcrumb
ion-breadcrumbs

Button
ion-button
ion-ripple-effect

Card
ion-card
ion-card-content
ion-card-header
ion-card-subtitle

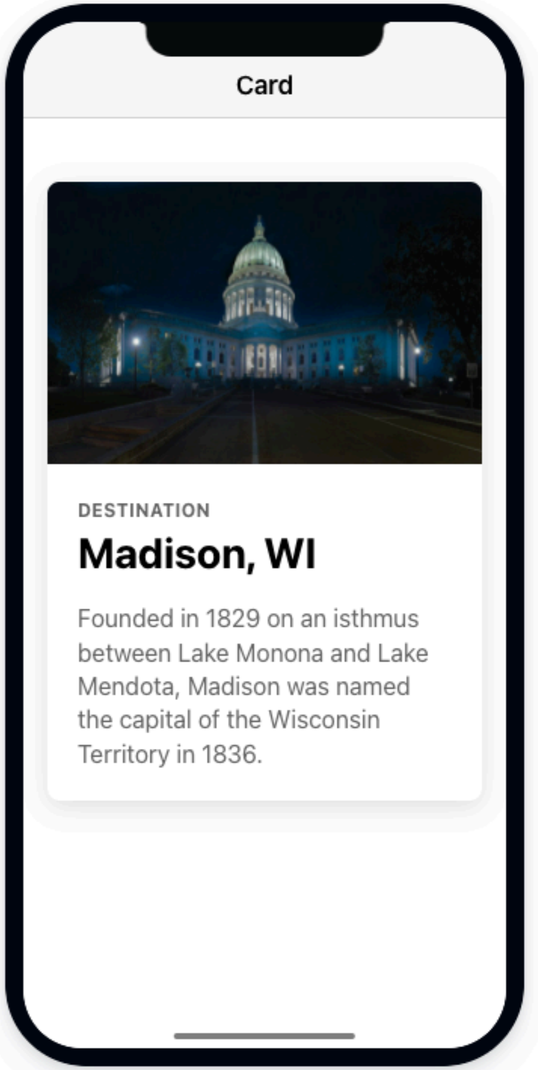
Usage

Angular | Javascript | React | Stencil | Vue

Copy

```
import React from 'react';
import { IonContent, IonHeader, IonPage, IonTitle, IonToolbar, IonCard, IonCard
import { pin, wifi, wine, warning, walk } from 'ionicons/icons';

export const CardExamples: React.FC = () => {
  return (
    <IonPage>
      <IonHeader>
        <IonToolbar>
          <IonTitle>CardExamples</IonTitle>
        </IonToolbar>
      </IonHeader>
      <IonContent>
        <IonCard>
          <IonCardHeader>
            <IonCardSubtitle>Card Subtitle</IonCardSubtitle>
            <IonCardTitle>Card Title</IonCardTitle>
          </IonCardHeader>

          <IonCardContent>
            Keep close to Nature's heart... and break clear away, once in awhil
            and climb a mountain or spend a week in the woods. Wash your spirit
      </IonCardContent>
        </IonCard>

        <IonCard>
          <IonItem>
            <IonIcon icon={pin} slot="start" />
            <IonLabel>ion-item in a card, icon left, button right</IonLabel>
            <IonButton fill="outline" slot="end">View</IonButton>
          </IonItem>

          <IonCardContent>
            This is content, without any paragraph or header tags,
            within an ion-cardContent element.
```

iOS | Android



TypeScript for JavaScript Programmers

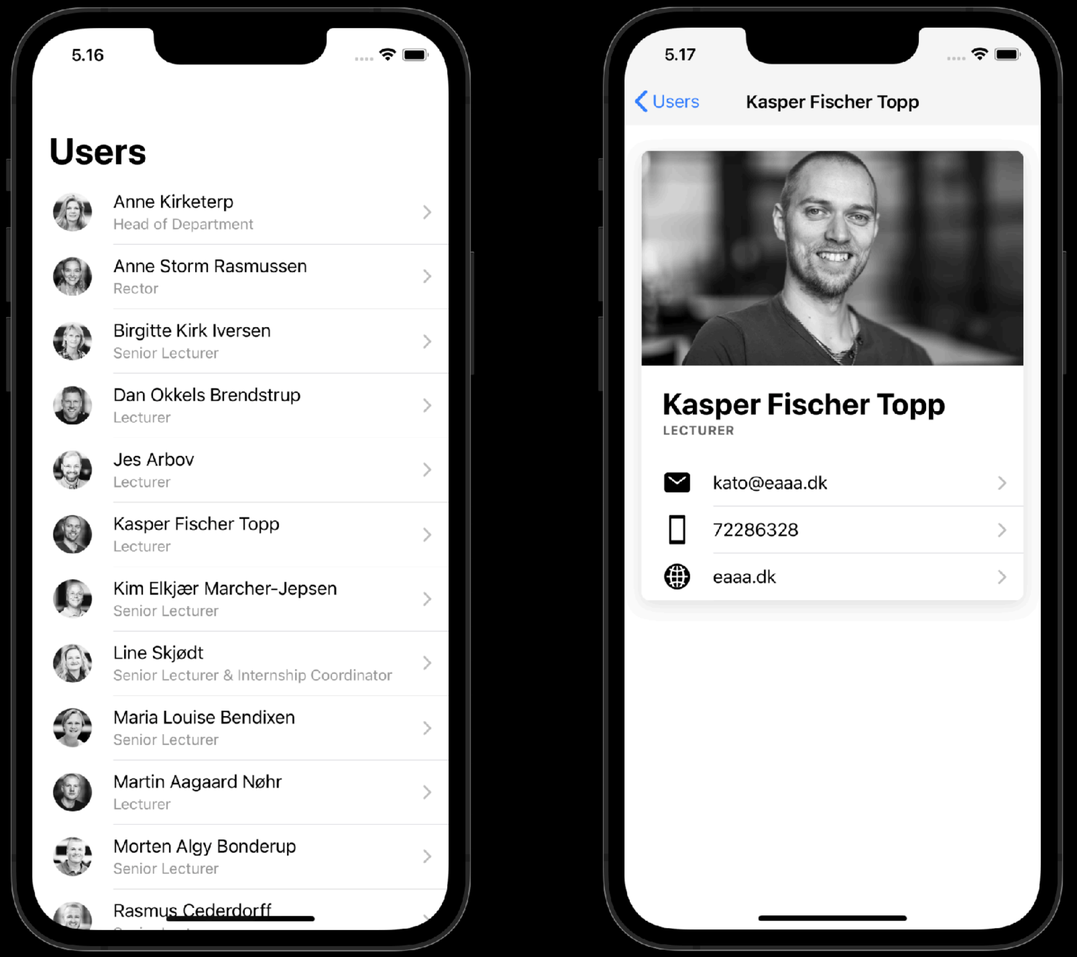
https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html

Ionic User List App

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

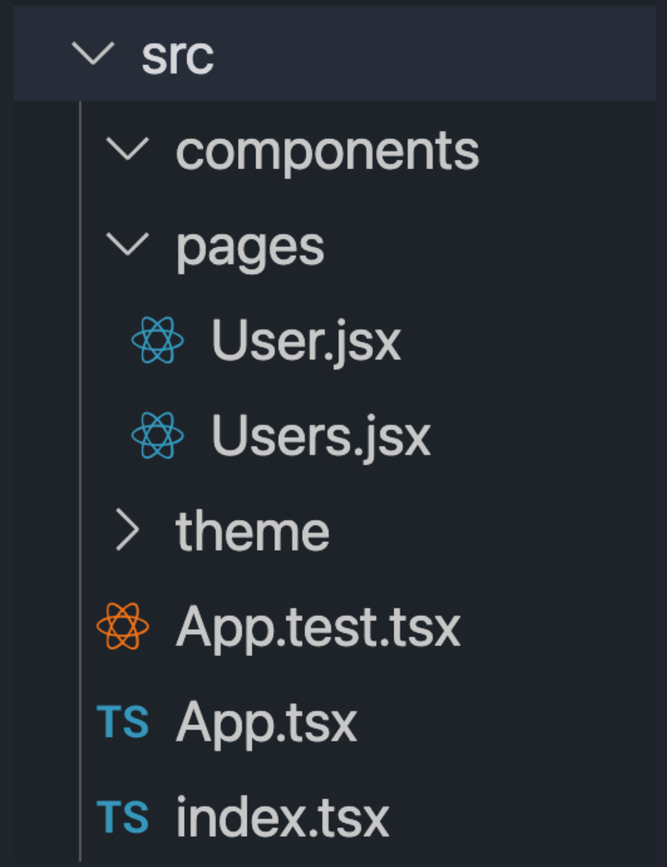
ionic-user-list

ionic-user-list-no-ts

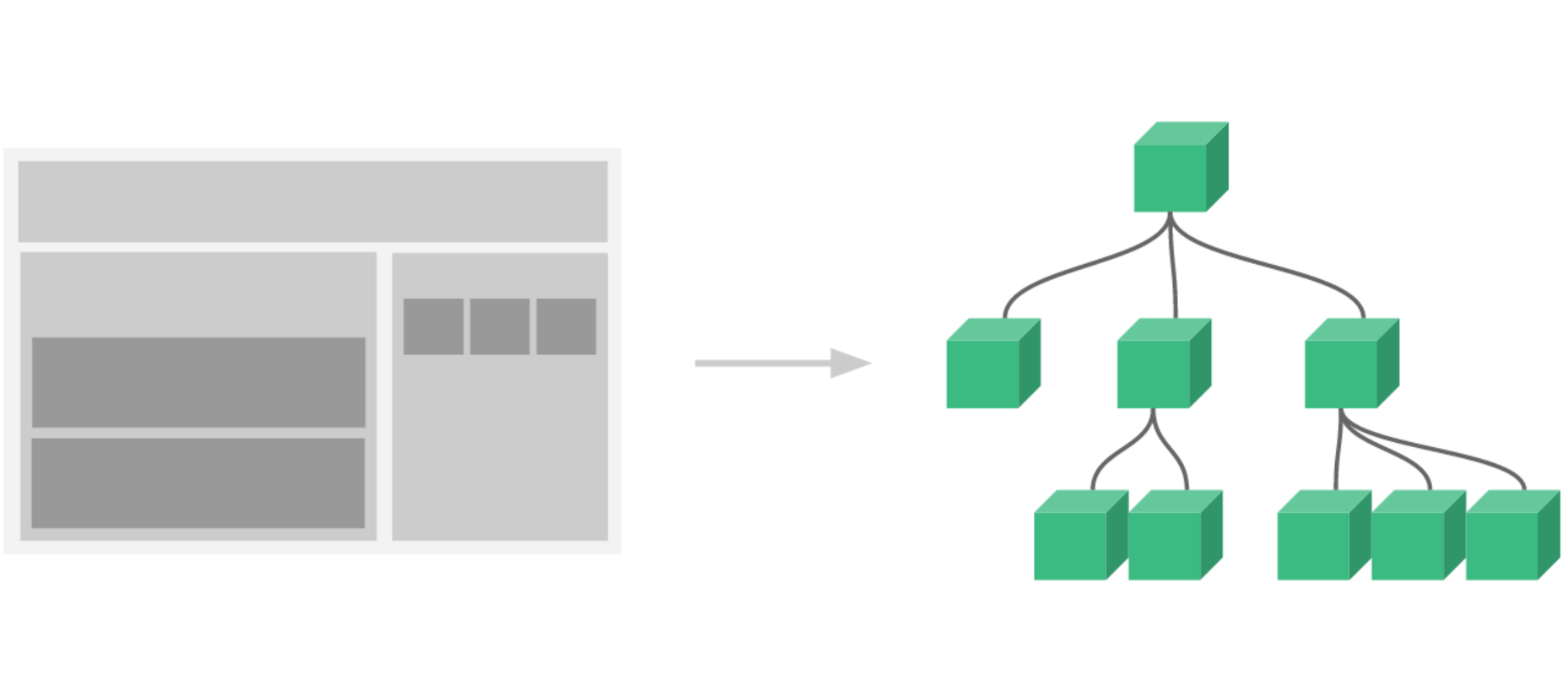


Thinking in React?

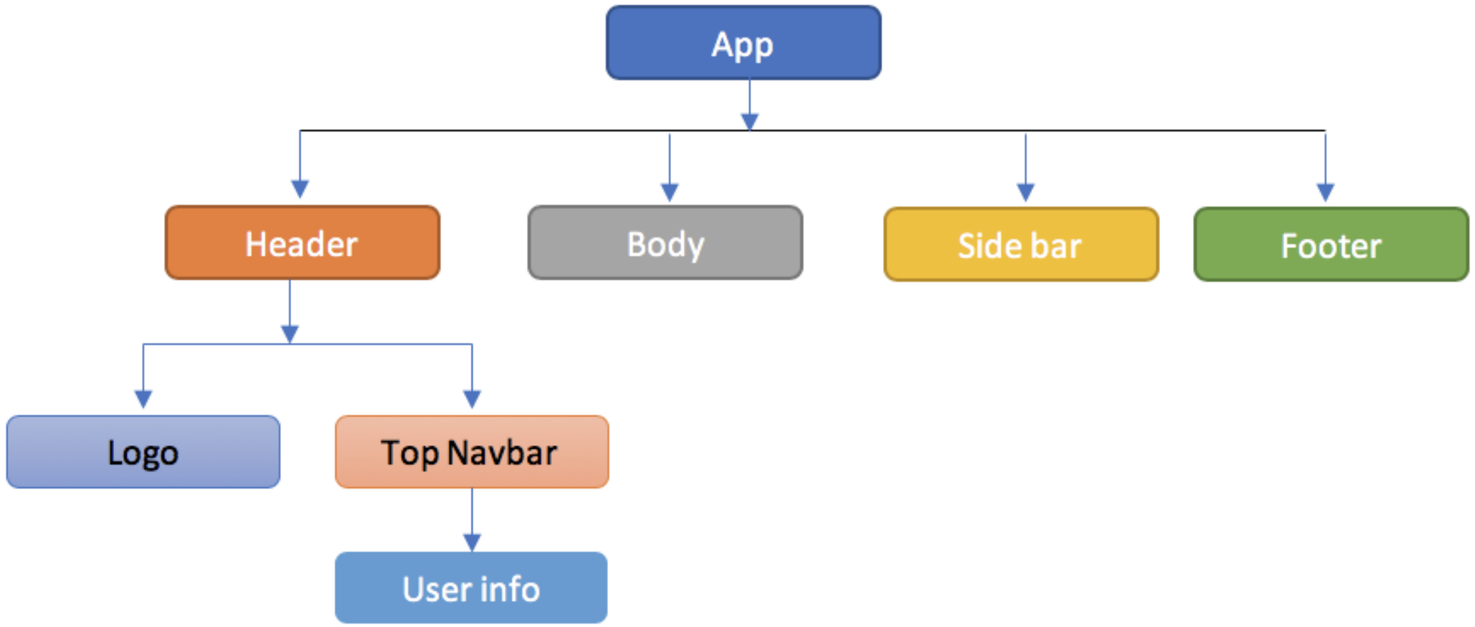
https://beta.reactjs.org/learn/thinking-in-react



COMPONENTS



APP STRUCTURED IN COMPONENTS



Structure

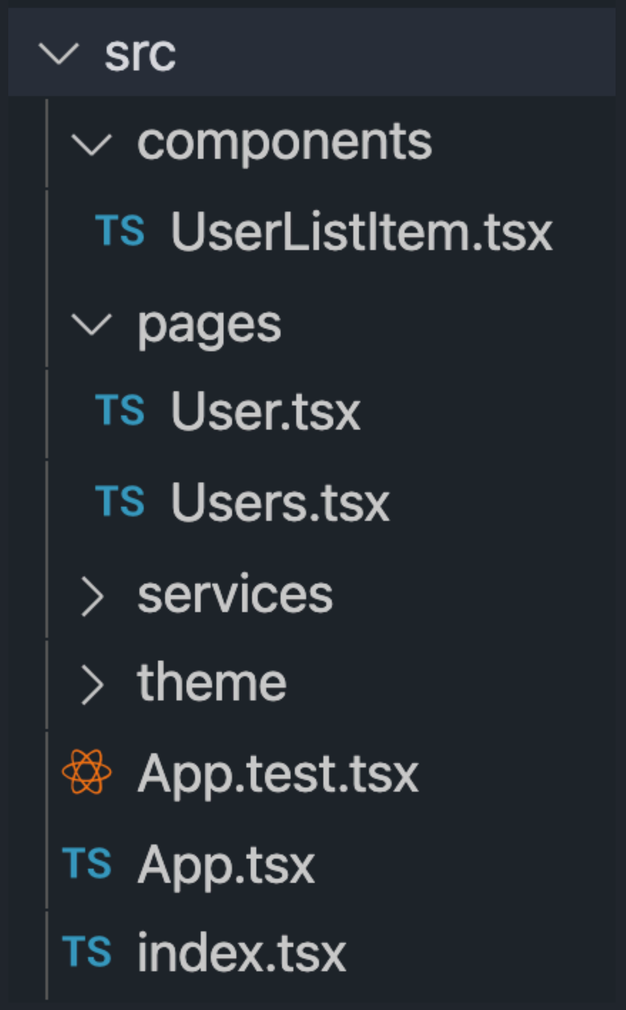
It’s all components

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

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

```tsx
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>
```

Components, components, components

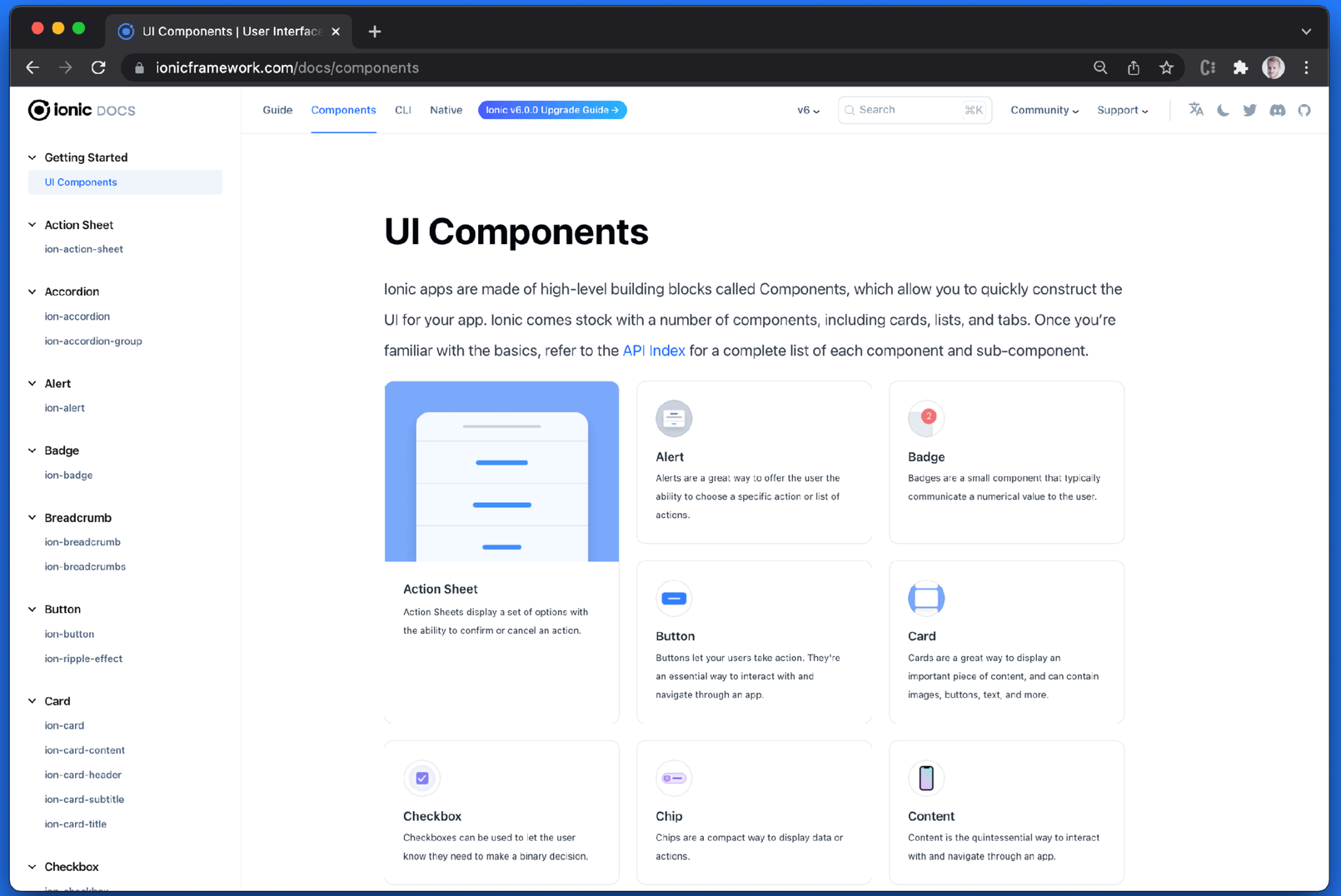
1. Review your Users component.
2. Think of how to restructure the component and divide it into several components.
3. Create a file and a new component named UserListItem or whatever you prefer.
4. Extract the IonItem to the newly created component. Refactor what’s needed.
5. Import and use the new component in Users and make it pass needed props.



Ionic Core Concepts

https://ionicframework.com/docs/core-concepts/fundamentals

UI Components



CARD SUBTITLE

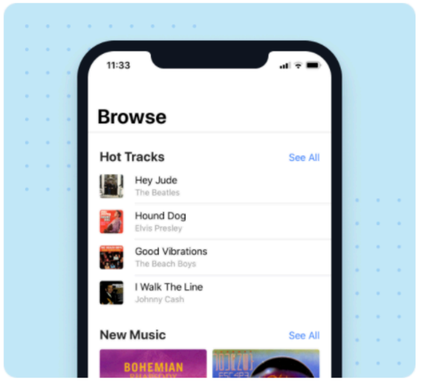
Card Title

Here's a small text description for the card content. Nothing more, nothing less.

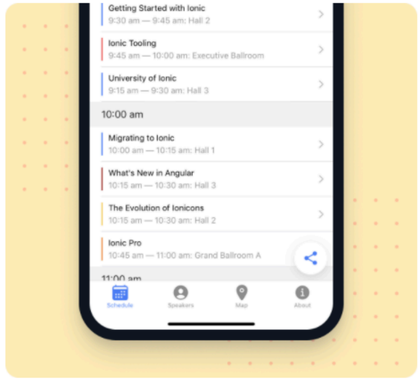
Action

Predefined UI Components

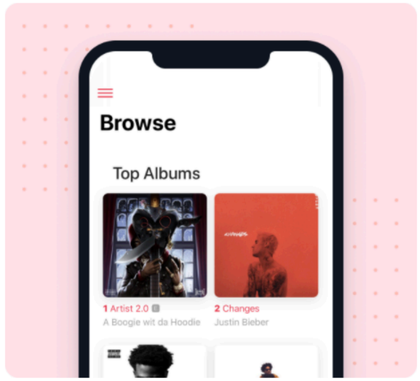
Awesome docs with many examples and sample apps



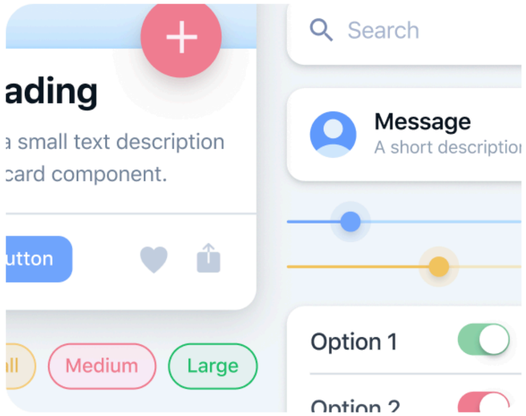
Music Player



Conference App



StarTrack



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.

```
$ ionic start
? Project name: My App
? Starter template: tabs
Success!
ionic run ios
Compiled successfully.
Deployed to iOS!
```

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.

Lifecycle of ion (React) Component

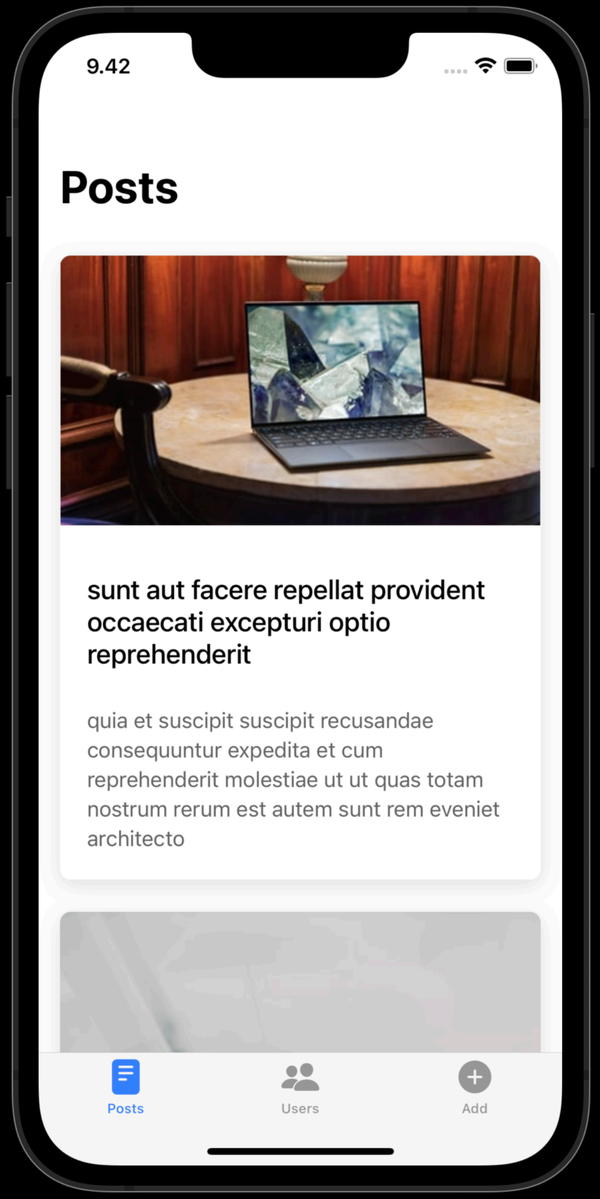
Lifecycle hooks: https://ionicframework.com/docs/react/lifecycle

Why lifecycle methods?

https://ionicframework.com/docs/react/lifecycle#lifecycle-methods-in-functional-components

1. Reimpliment Users and User component using one of the Ionic React Lifecyle Hooks.

2. Consider the use and differences between `useEffect(…)` and `useIonViewWillEnter(…)`.



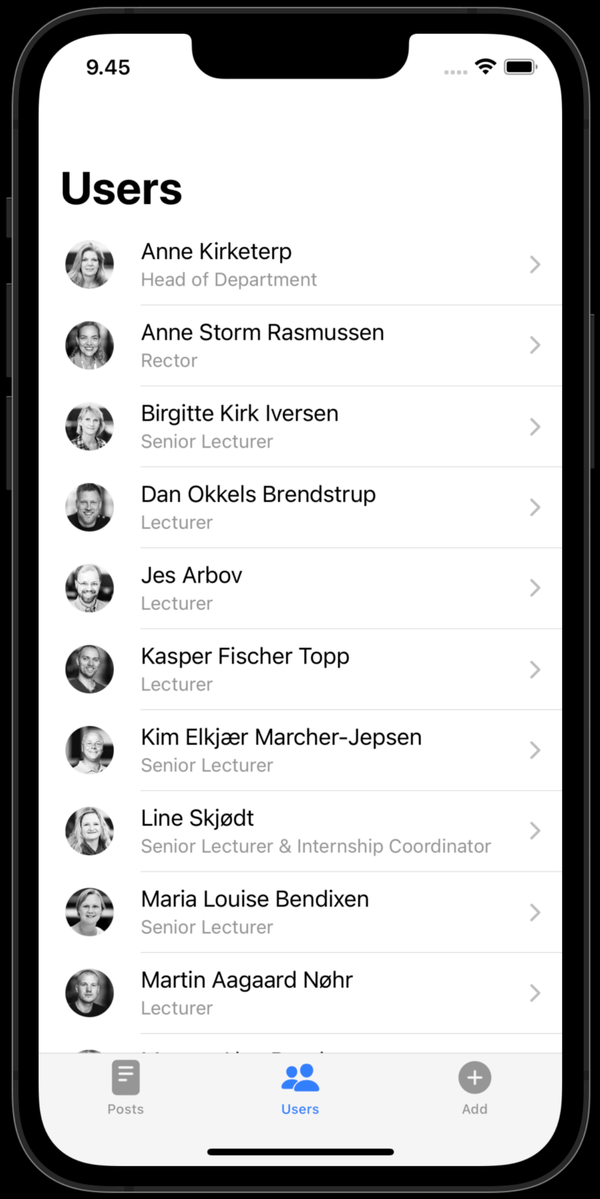
Ionic Post App

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

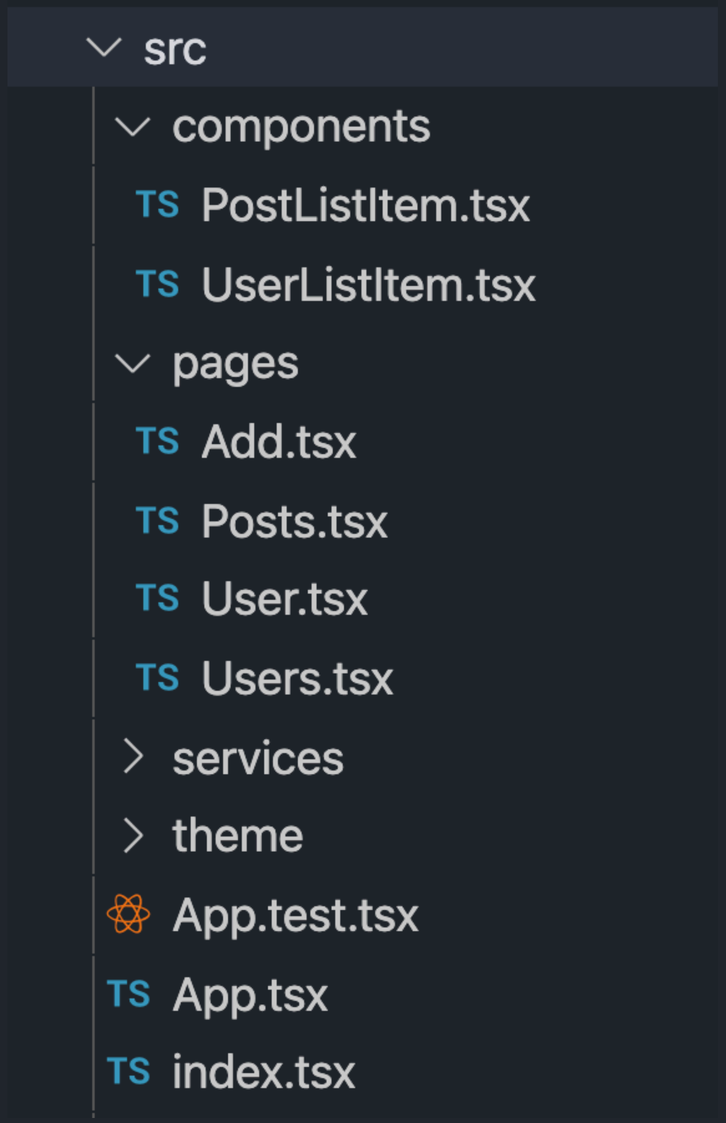
5. Add the list of users in the second tab, Users Page.

6. Make sure you can navigate to the detail view.

Ionic Post App

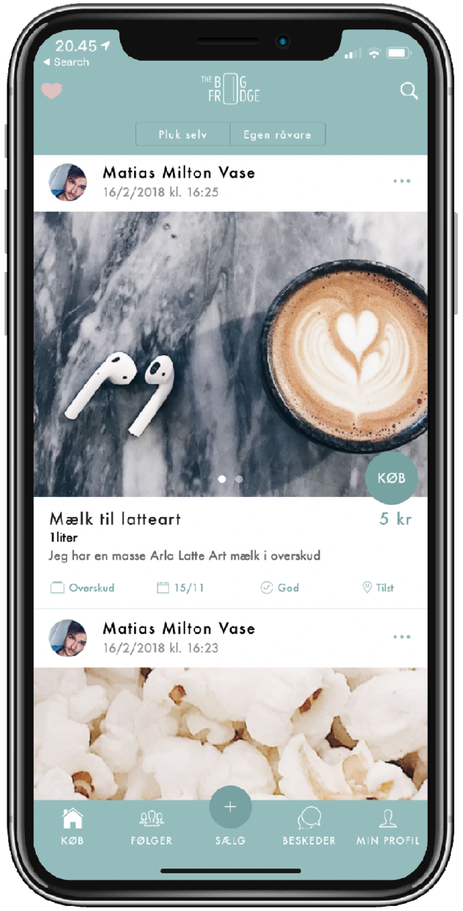
7. Think in components. Is there any logic that might be useful to extract to separate components?

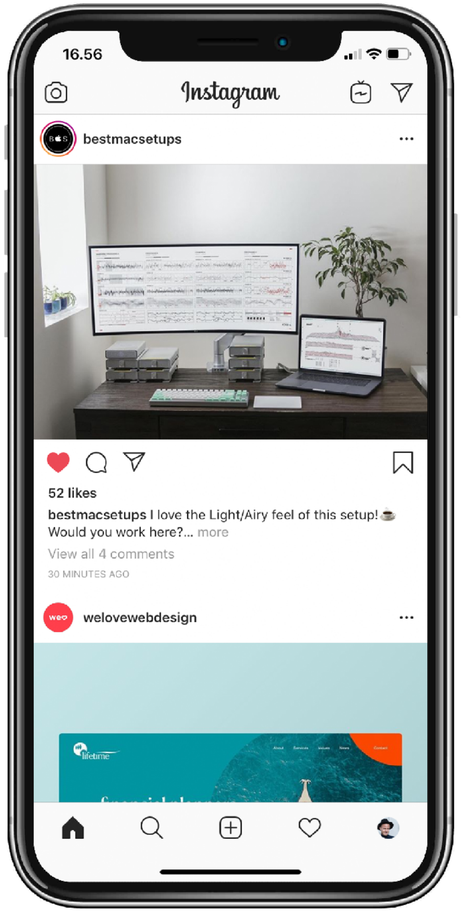
8. Reimplement your pages using components.



Display User

... on given post (uid)





Ionic Post App

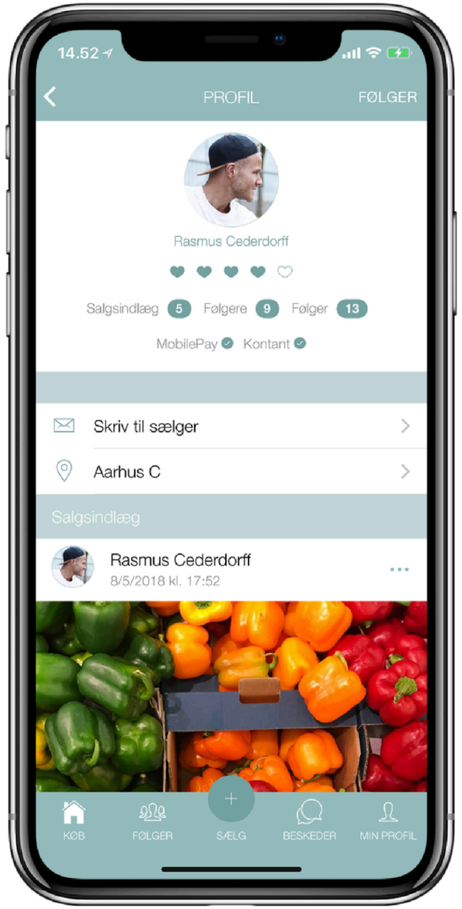
9. Improve your post view by adding user details (post create by...). Start by adding hardcoded user details like user avatar and user name using Ionic UI Components.

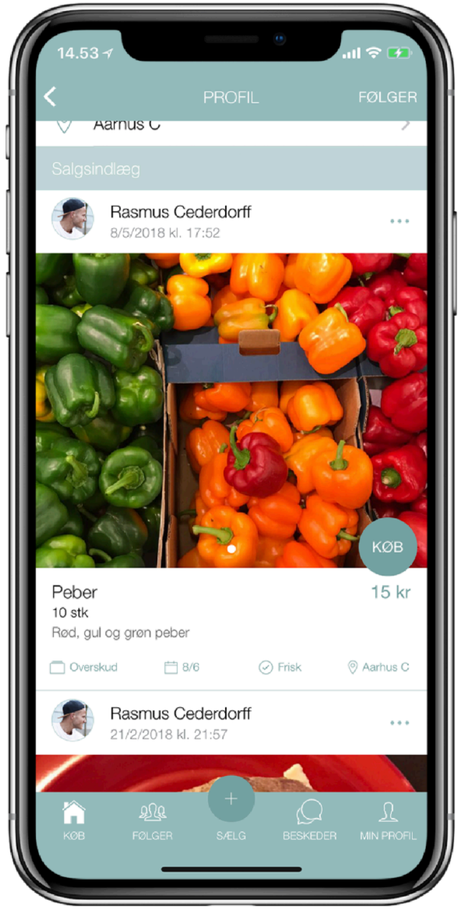
10. Implement logic to concatenate the posts array with user data from users array. All post objects have a `uid` property.

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.

Navigation & Routing

https://ionicframework.com/docs/react/navigation

ion-router

https://ionicframework.com/docs/api/router

ion-nav

https://ionicframework.com/docs/api/nav

Add New Page

1. Add a new function component (a new page).

2. Add a new route in App.tsx to handle routing of your new page.

3. Test and make sure the navigation works as expected.

4. Next, change the redirect so your new page is the default page of your app.

Types of Mobile Navigation

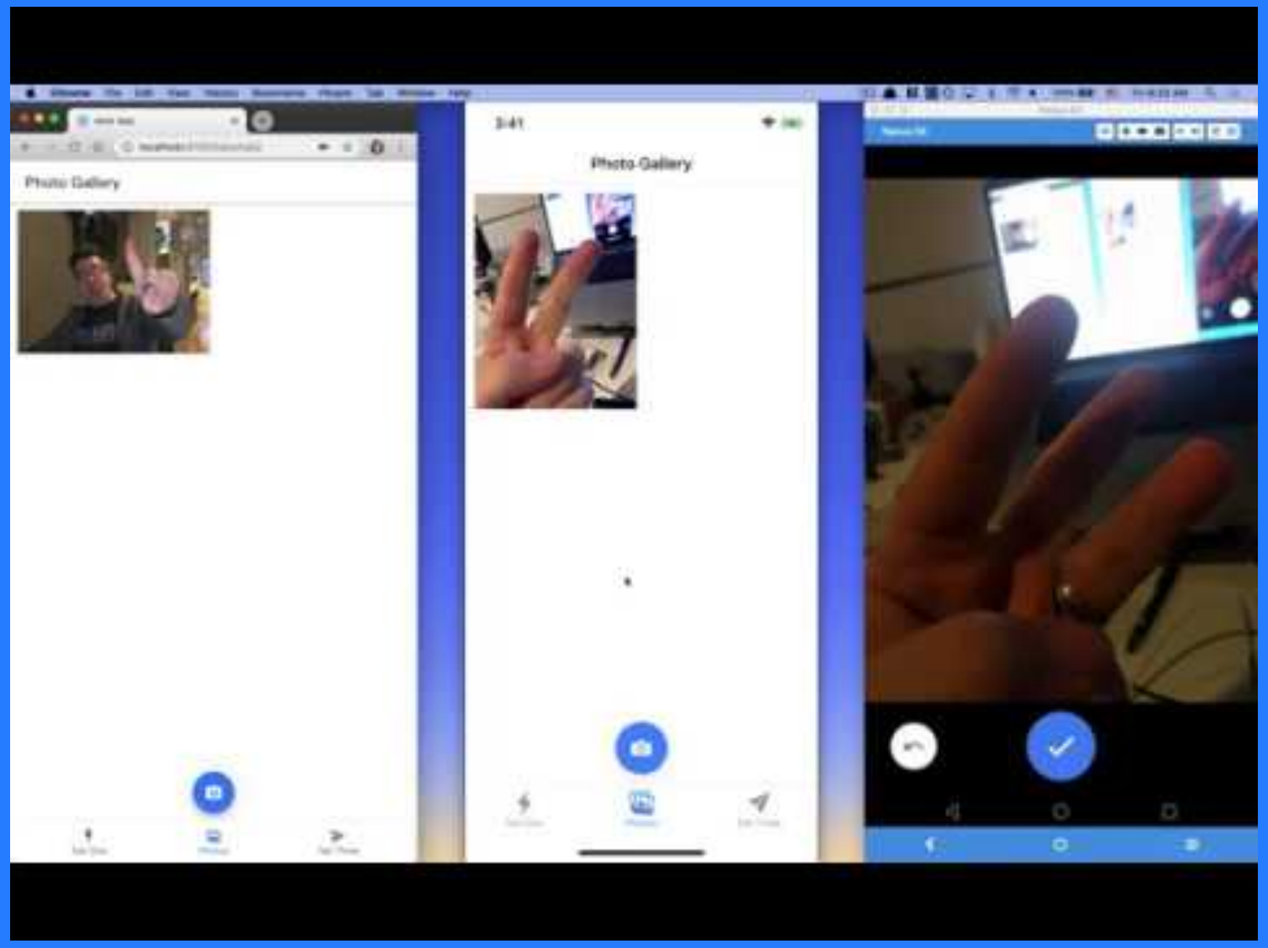
What kind of types do you know?

Add a modal

https://ionicframework.com/docs/api/modal#usage

Ionic Camera App

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



Next Thursday