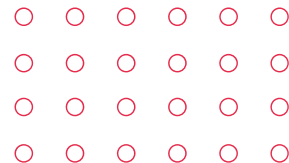


# Just Another React Introduction

Santosh Muthukrishnan & Goh Jun Yi



Slides PDF



# Today's Plan

01

## React Intro

Learn about React, its capabilities and structure

02

## React Libraries

Specifically React-Router & Redux

03

## React Demo

Let's build a simple app with React

04

## Break / QnA

05

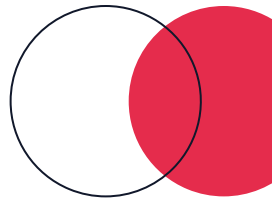
## Source Academy

SA's specific tech stack

06

## Solve A Bug!

A demo through SA's tech stack





# 01

## React



# React, Why?



## React Components

Take advantage of  
abstraction



## High Performance

Virtual DOM in memory  
minimise read/writes



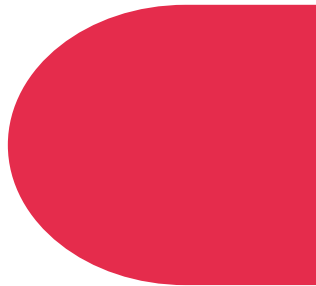
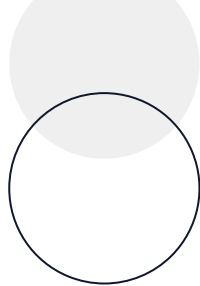
## Supporting Libraries

Many 3<sup>rd</sup> party libraries  
are available



## Growing Community

More support is always  
great



# Characteristics of React



## Unopiniated

Built and structure of the application is your discretion



## Lightweight

Expand your application with tools when neccessary



## Declarative

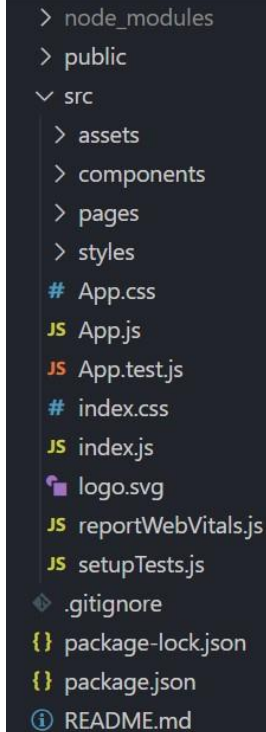
React does the magic for you!



# React Project Structure

React is an *unopiniated* language, hence the project structure is up to the developer's discretion. Every react project can look different in structure.

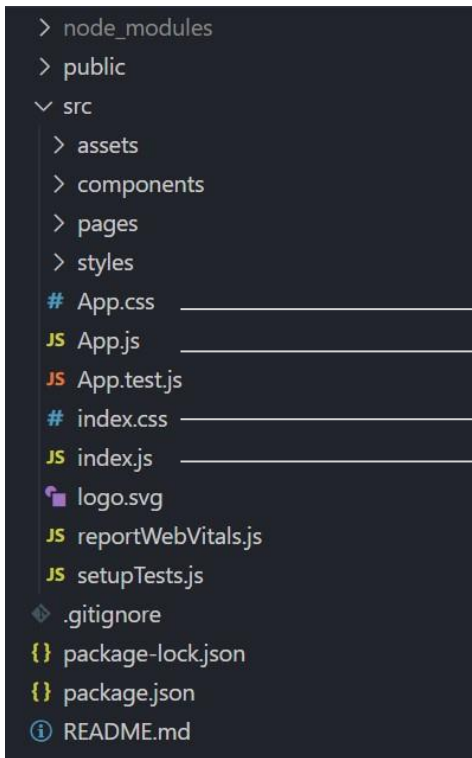
However, there are some typical project structure that React developers have adopted so let's take a look at one.



```
> node_modules
> public
▼ src
  > assets
  > components
  > pages
  > styles
  # App.css
  JS App.js
  JS App.test.js
  # index.css
  JS index.js
  logo.svg
  JS reportWebVitals.js
  JS setupTests.js
.gitignore
{} package-lock.json
{} package.json
(i) README.md
```

*Disclaimer: The project in the image above was created using Create-React-App, while folders in src/ were manually created*

# React Project Structure

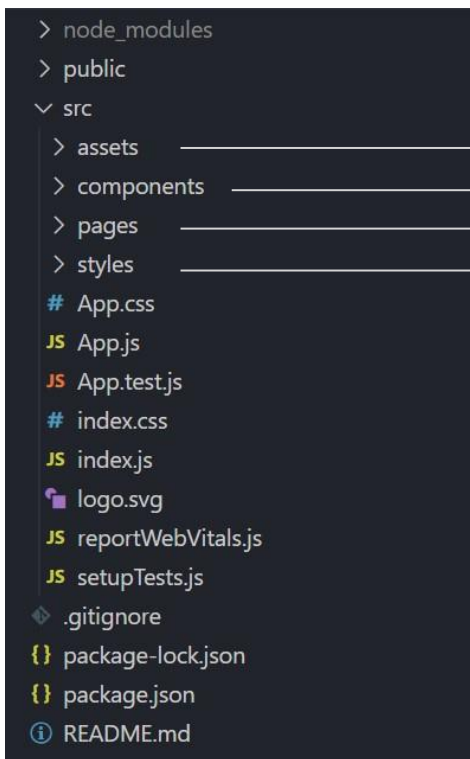


## src/ folder

- App.css → App.js specific stylesheet
- App.js → Top level react component
- index.css → Global stylesheet
- index.js → Entry point of application

Disclaimer: The project in the image above was created using Create-React-App, while folders in src/ were manually created

# React Project Structure



## src/ folder

- media files (e.g. images, icons)
- React components (eg. popup-card, app-button)
- Pages that users navigates between (eg. info-page, main-page)
- Stylesheets (eg. .css / .scss files)

Other folders in src/ may be created for other use cases or libraries.

Eg. React hooks would usually share hooks in a hooks/ folder in src/

*Disclaimer: The project in the image above was created using Create-React-App, while folders in src/ were manually created*

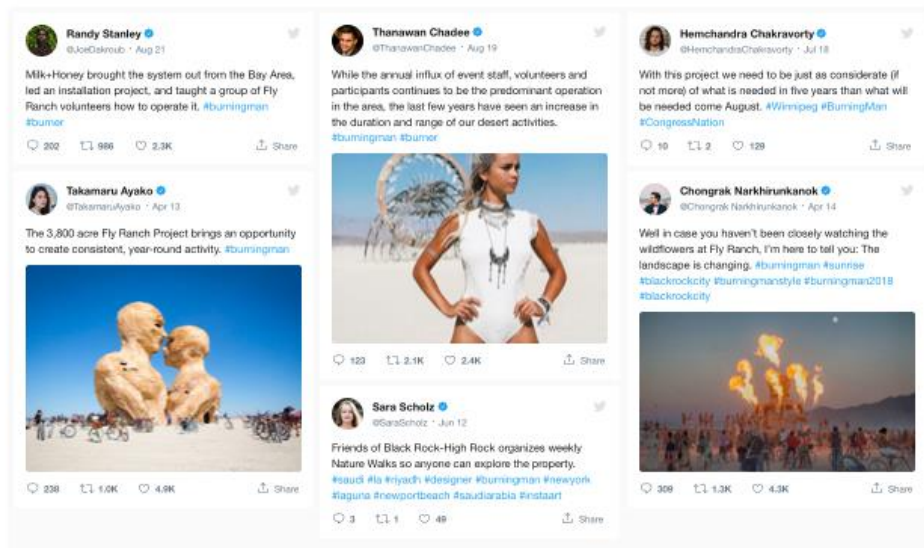


# React Components

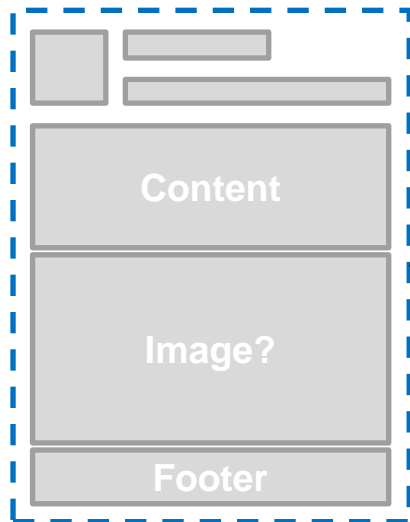
React components are building blocks for a React application.

Components are like **blueprints**, allowing you to recreate the same object multiple times anywhere in your application.

We can take advantage of **abstraction**.



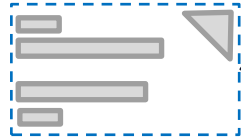
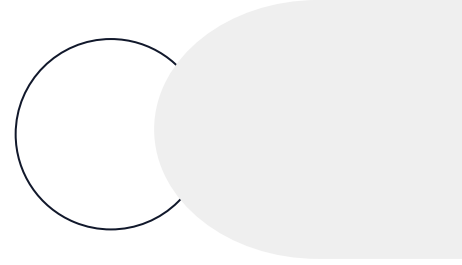
# React Components



Tweet Component



# React Components



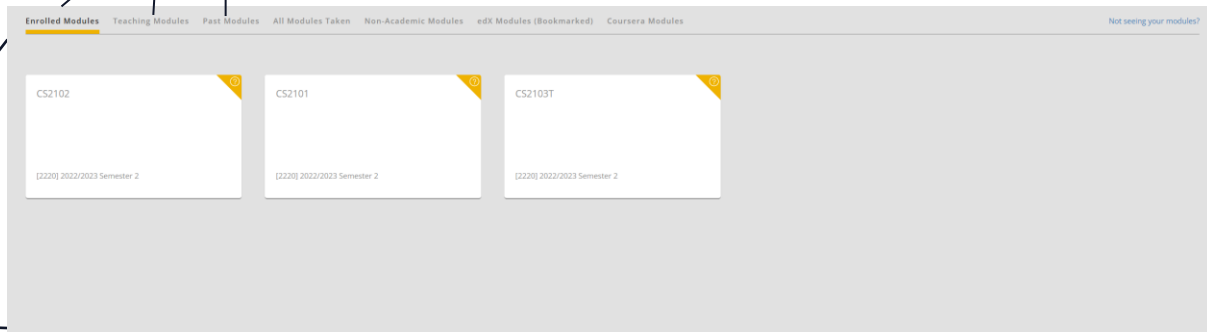
**Module Card  
Component**

<div>CS2101</div> <div>Efftv Comm for Computg Pro</div> <div>[2220] 2022/2023 Semester 2</div> <div>Modular Credit: 4</div>	<div>CS2102</div> <div>Database Systems</div> <div>[2220] 2022/2023 Semester 2</div> <div>Modular Credit: 4</div>	<div>CS2103T</div> <div>Software Engineering</div> <div>[2220] 2022/2023 Semester 2</div> <div>Modular Credit: 4</div>	<div>CFG1002</div> <div>Career Catalyst</div> <div>[2210] 2022/2023 Semester 1</div> <div>Modular Credit: 2</div>	<div>CS2030S</div> <div>Programming Methodology II</div> <div>[2210] 2022/2023 Semester 1</div> <div>Modular Credit: 4</div>
<div>CS2106</div> <div>Intro to Operating Systems</div> <div>[2210] 2022/2023 Semester 1</div> <div>Modular Credit: 4</div>	<div>CS2109S</div> <div>Introduction to AI and Machine</div> <div>[2210] 2022/2023 Semester 1</div> <div>Modular Credit: 4</div>	<div>GEC101S</div> <div>Public Health in Action</div> <div>[2210] 2022/2023 Semester 1</div> <div>Modular Credit: 4</div>	<div>CP3107</div> <div>Computing for Voluntary Welfar</div> <div>[2140] 2021/2022 Special Term(Part 2)</div> <div>Modular Credit: 6</div>	<div>CP3107</div> <div>Computing for Voluntary Welfar</div> <div>[2130] 2021/2022 Special Term(Part 1)</div> <div>Modular Credit: 6</div>
<div>CS1010R</div> <div>Programming Methodology</div> <div>[2120] 2021/2022 Semester 2</div> <div>Modular Credit: 1</div>	<div>CS2040S</div> <div>Data Structures and Algorithms</div> <div>[2120] 2021/2022 Semester 2</div> <div>Modular Credit: 4</div>	<div>CS2100</div> <div>Computer Organisation</div> <div>[2120] 2021/2022 Semester 2</div> <div>Modular Credit: 4</div>	<div>ES2660</div> <div>Communicating in the Informati</div> <div>[2120] 2021/2022 Semester 2</div> <div>Modular Credit: 4</div>	<div>GEA1000</div> <div>Quantitative Reasoning</div> <div>[2120] 2021/2022 Semester 2</div> <div>Modular Credit: 4</div>

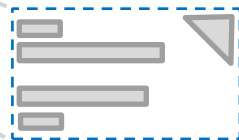


# React Components

Multiple tabs with same layout



Module View  
Component



Module Card  
Component



# React Components

## Functional

Functions that return a JSX / React element

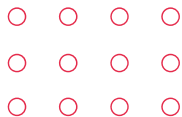
```
function AnExampleFunctionComponent(props) {  
  return <p>Welcome to {props.website_name}</p>;  
}
```

## Class

ES6 Class that extends upon React component class

```
class AnExampleClassComponent extends React.Component {  
  render() {  
    return <p>Welcome to {props.website_name}</p>;  
  }  
}
```

Since the introduction of *React Hooks*, Functional components seems to be the **recommended** choice by React Devs.





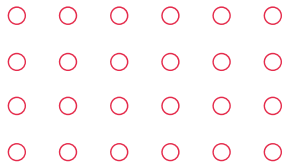
# 02

# React Libraries

React Router & Redux



# React Router



React Router enables **client-side routing** for your React application.

When loading a new page, the browser usually sends a request to retrieve the data/layout for the page.

With React router, loading a page from the same React application will only trigger React to decide what components needs to be rendered or removed from view.



# React Router



Hi, how can we help?

Search how-tos and more



[Guest](#) [Host](#) [Experience Host](#) [Travel admin](#)

We're here for you

Log in to get help with your reservations, account, and more.

Log in or sign up

Guides for getting started

[Browse all topics >](#)



Getting started with Airbnb



Accessing your account



Help with a reservation

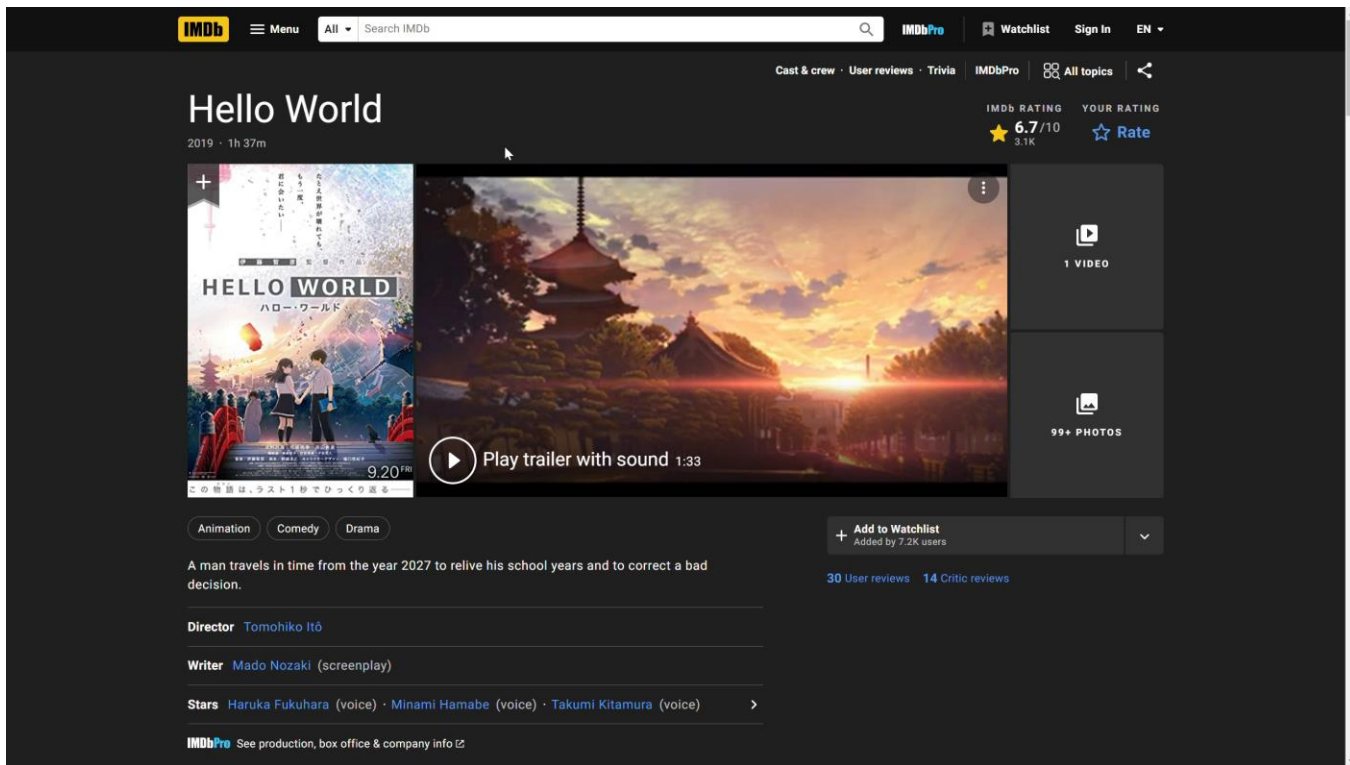


Getting protected through  
AirCover





# React Router



# React Router

Help Centre



Hi, how can we help?

Search how-tos and more



Guest Host Experience Host Travel admin

We're here for you

Log in to get help with your reservations, account, and more.

Log in or sign up

Guides for getting started

Browse all topics >



Getting started with Airbnb



Accessing your account



Help with a reservation

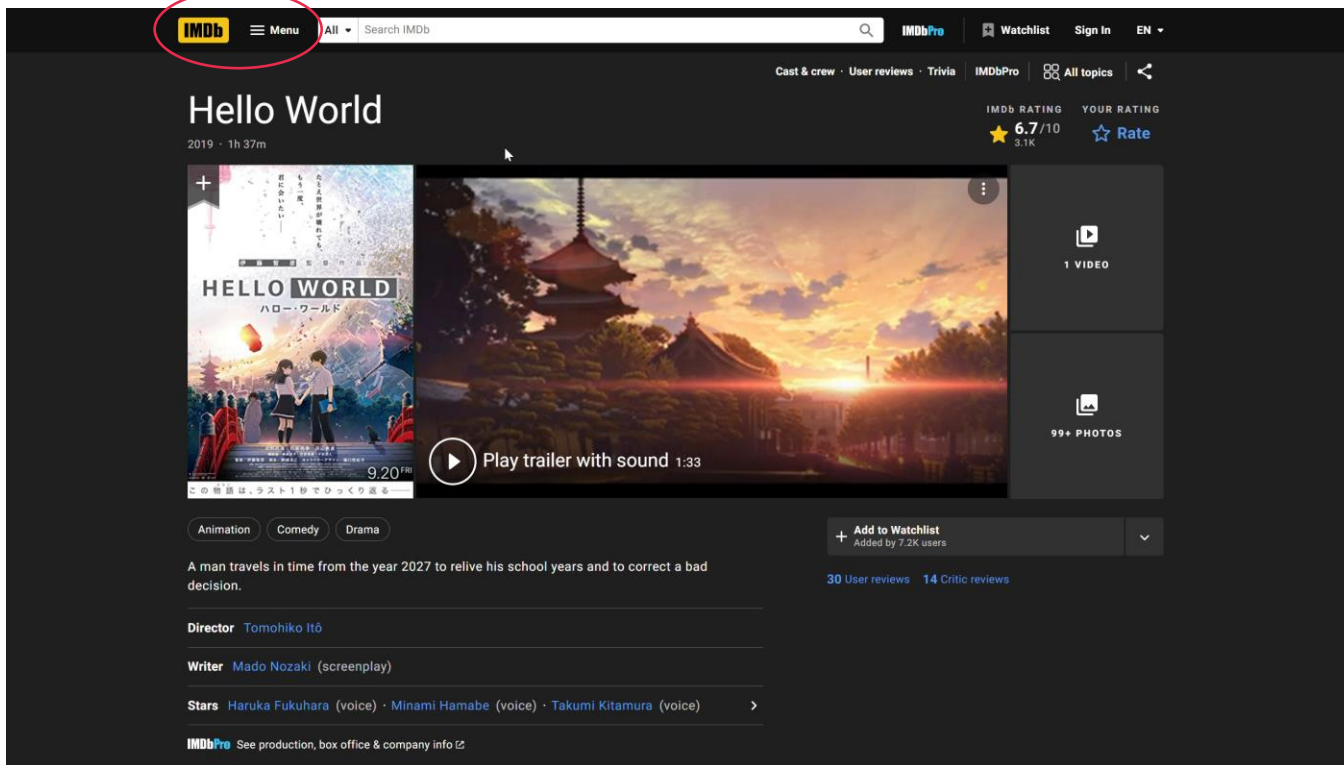


Getting protected through  
AirCover

Stays  
relatively  
static

Necessary  
content  
refreshes

# React Router



# React Router



## Pros

- Routing between components/pages are more likely to be quicker
- Animations and transitions between pages
- Does not incur a browser refresh (white screen flash)

## Cons

- Initial loading time is large, the whole website is loaded on initialisation (Code-splitting can circumvent this issue)
- Routing logic may be cumbersome to maintain
- Search engine crawling algorithms may not pick up site contents well

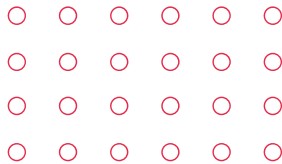
Side Note: **Server-side routing** is supposedly possible with React with external libraries



# Redux

Redux is a library that assists in **managing and updating global application state**, using events called 'actions'.

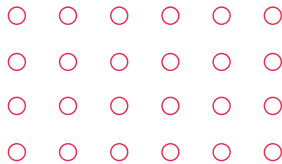
It serves as a **centralized store** for state that needs to be used across your entire application, with rules ensuring that the state can only be updated in a predictable fashion.



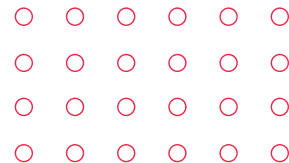
# Redux, why?




Class/Functional components can have states of their own, so what's the point?

Certain states are passed down from parent to children, and possibly **propagated deeply**. The codebase can grow increasingly unreadable / clunky.



# Redux, why?



Administration					Logged in: Darth Vader
Luke Skywalker					
Darth Vader					
Padmé Amidala					

Can only edit  
your own  
information



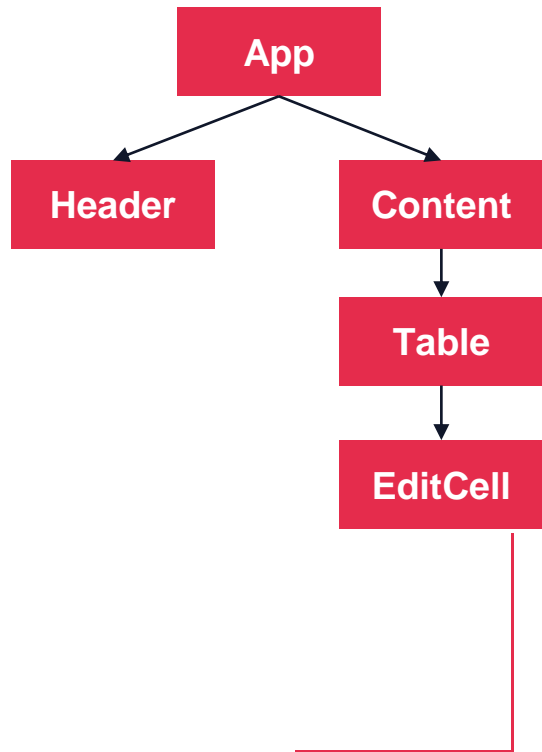
# Redux, why?



☰ Administration

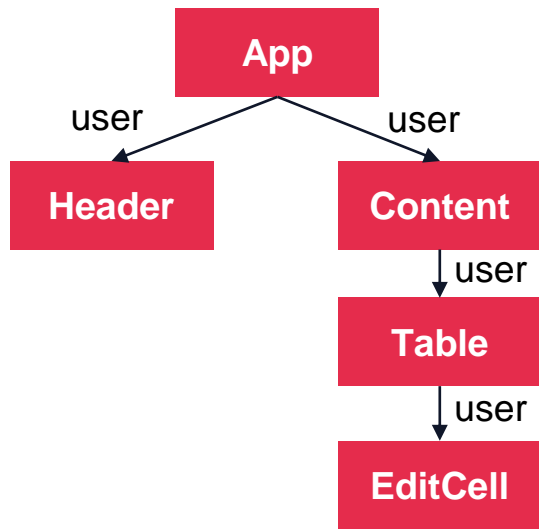
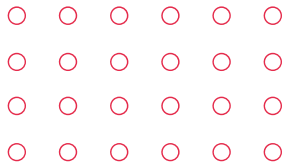
Logged in:  
Darth Vader

	Faction	...	Admin	Owner	
Luke Skywalker	Rebel		No	No	👁
Darth Vader	Dark		No	No	✎
Padmé Amidala	Rebel		No	No	👁



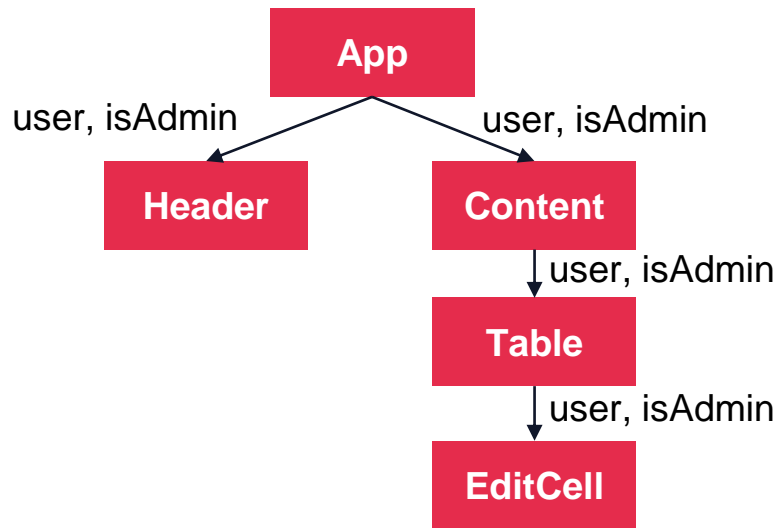


# Redux, why?

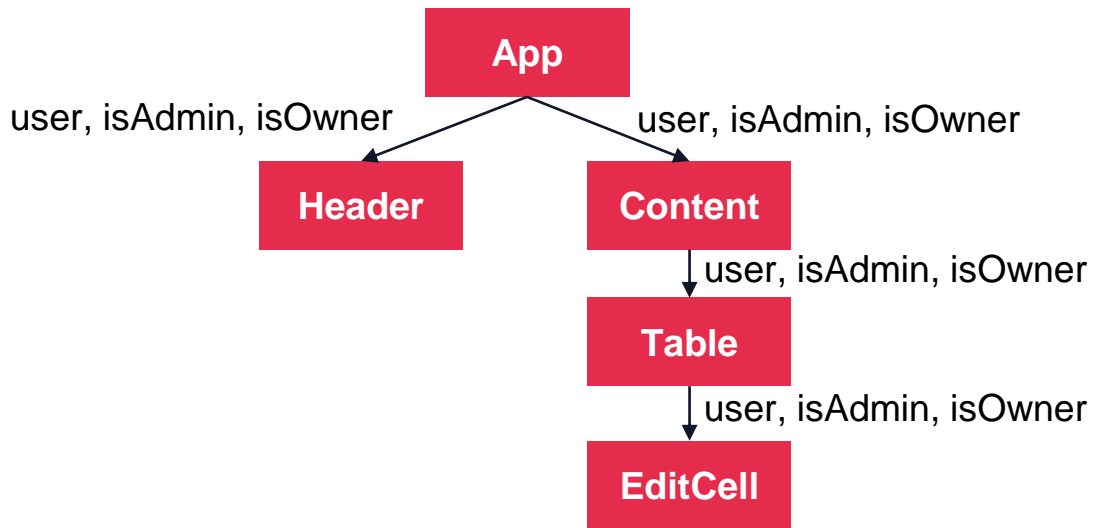
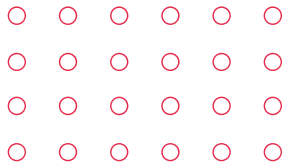


```
...  
row.name === user  
  ? <ViewButton />  
  : <EditButton />  
...
```

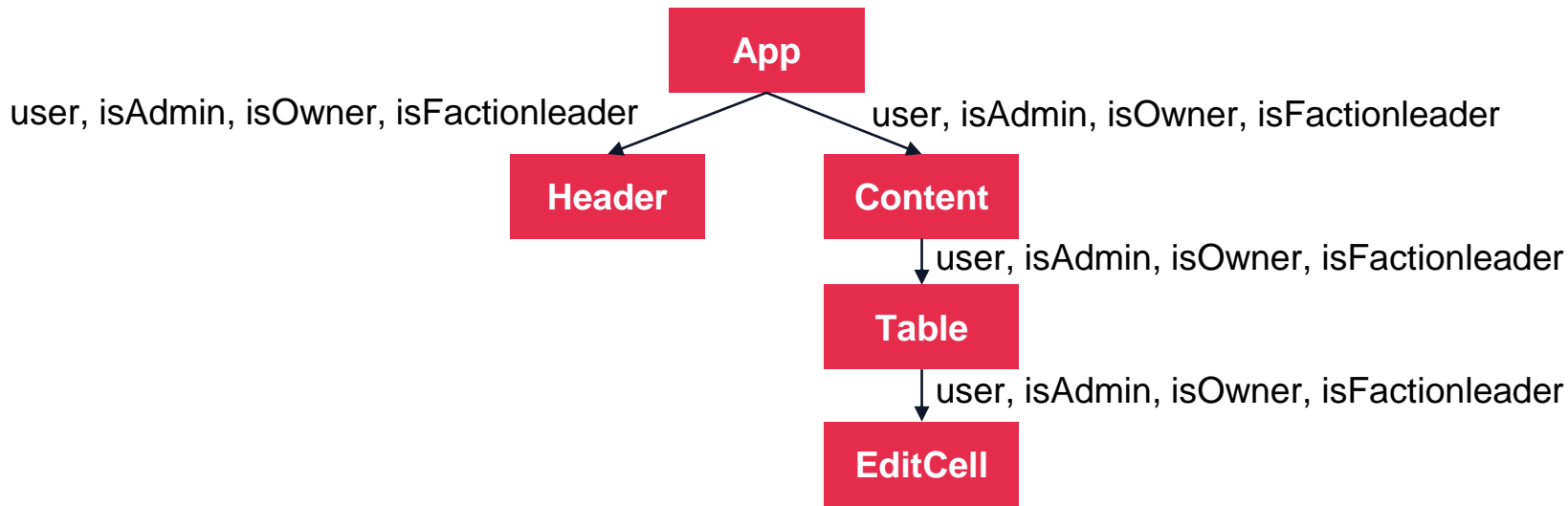
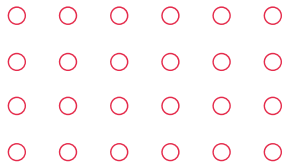
# Redux, why?



# Redux, why?



# Redux, why?



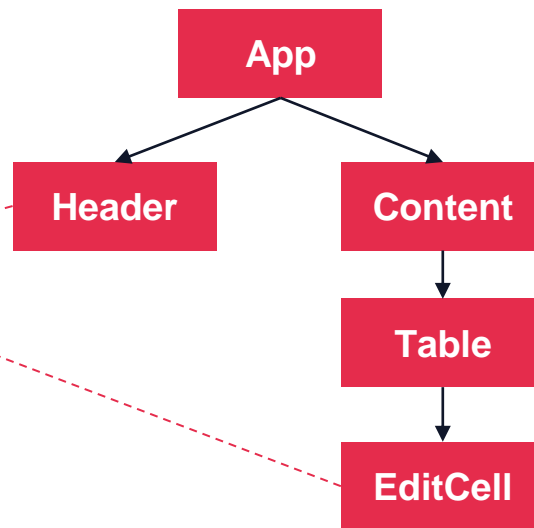
Surely there's a better way...

# Redux!

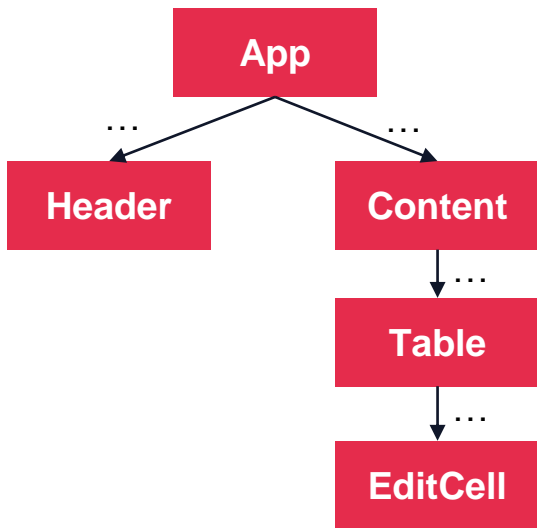
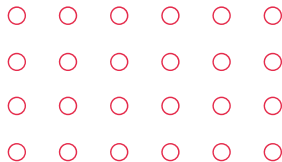


```
user: {  
  name: "Darth Vader",  
  isAdmin: False,  
  isOwner: False,  
  isFactionLeader: False,  
},  
...
```

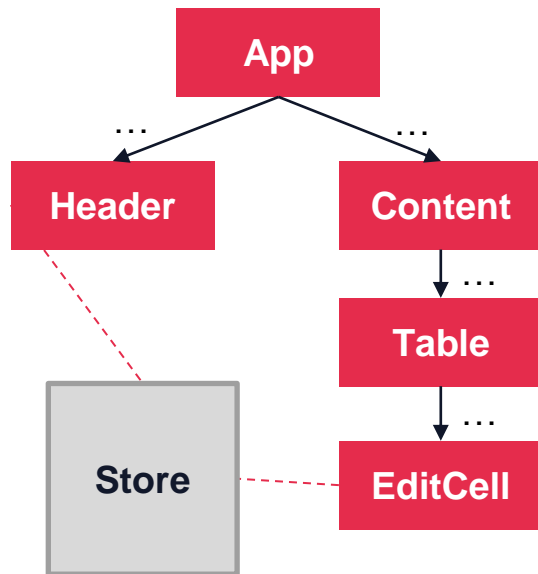
Redux Store



# Redux!



Propagate only



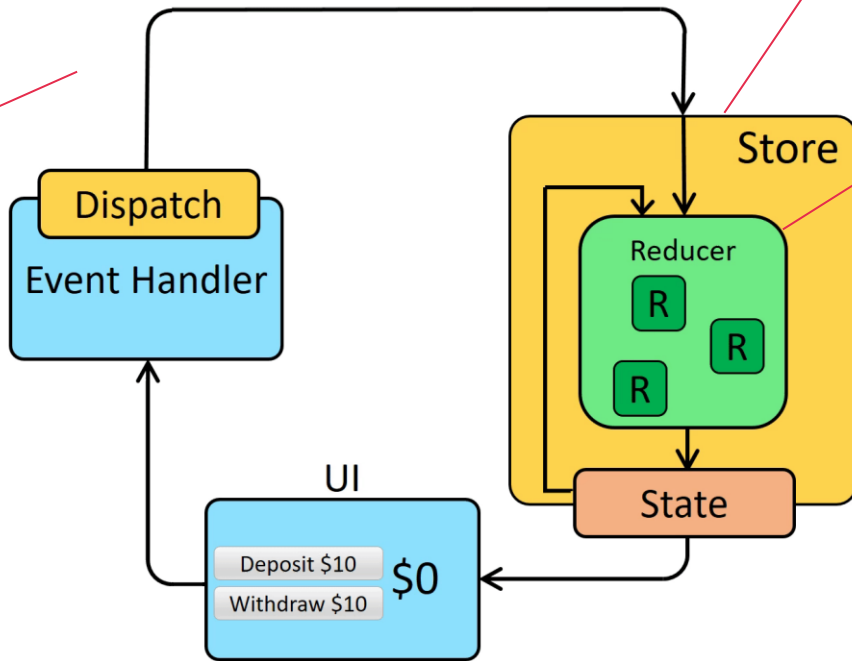
Propagate & Global Store

There are tradeoffs to use Redux, decide whether your project requires it!

# Redux, The Magic

## Action Object

Describes the type of action, and the data



## Store

Contains the reducers and states

## Reducer

Takes an action object and the previous state and **returns a new state**



03

# React Demo

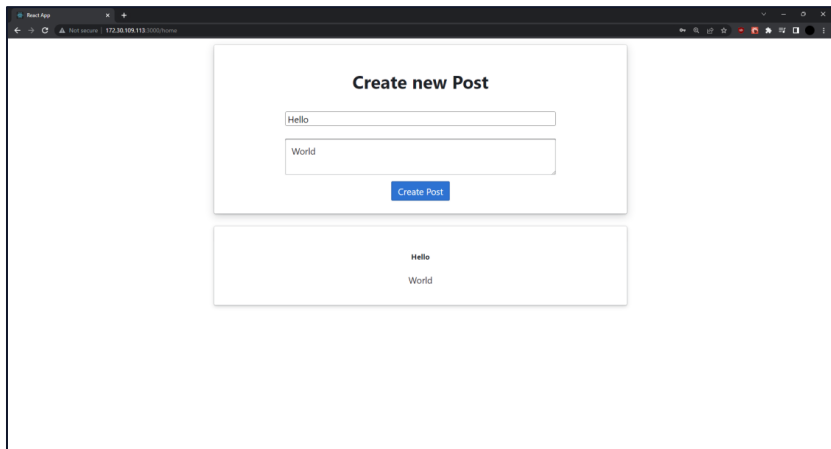
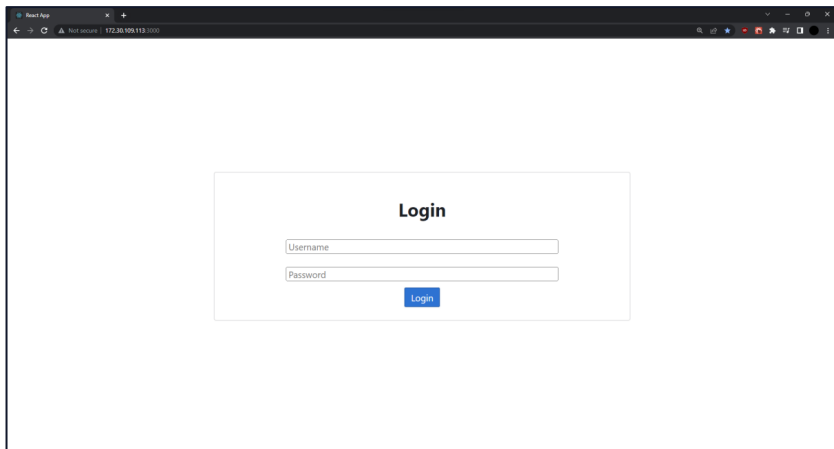




# What we're building today



- ❖ A very simple web app with a login page and a home page to create posts
- ❖ Covers basic React, BlueprintJs, Redux, React Router, Axios



# Installing React



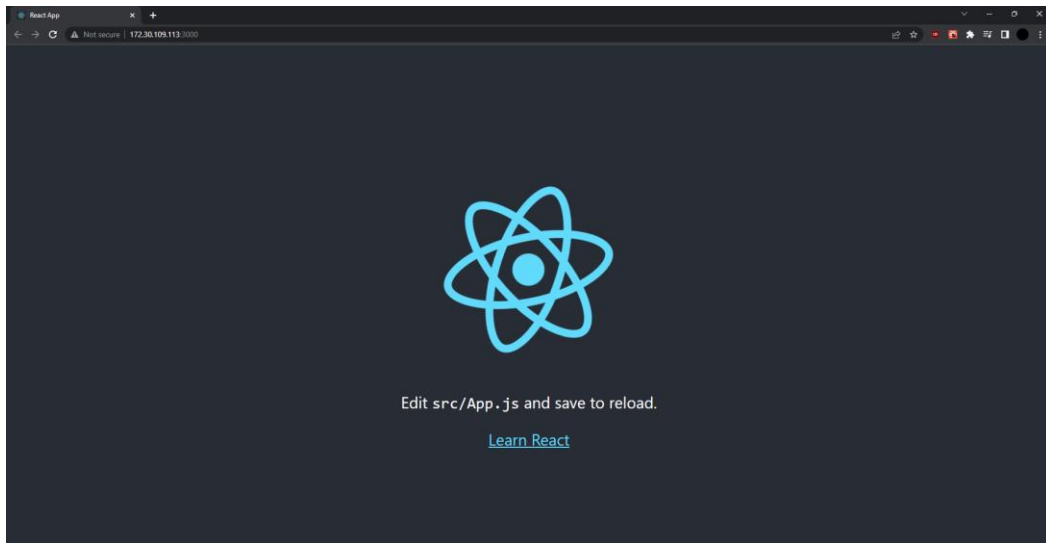
Installation Links:

- ❖ <https://medium.com/macoclock/install-react-js-on-mac-7cffe8bda2ac>
- ❖ <https://www.linuxtuto.com/how-to-install-reactjs-on-ubuntu-22-04/>

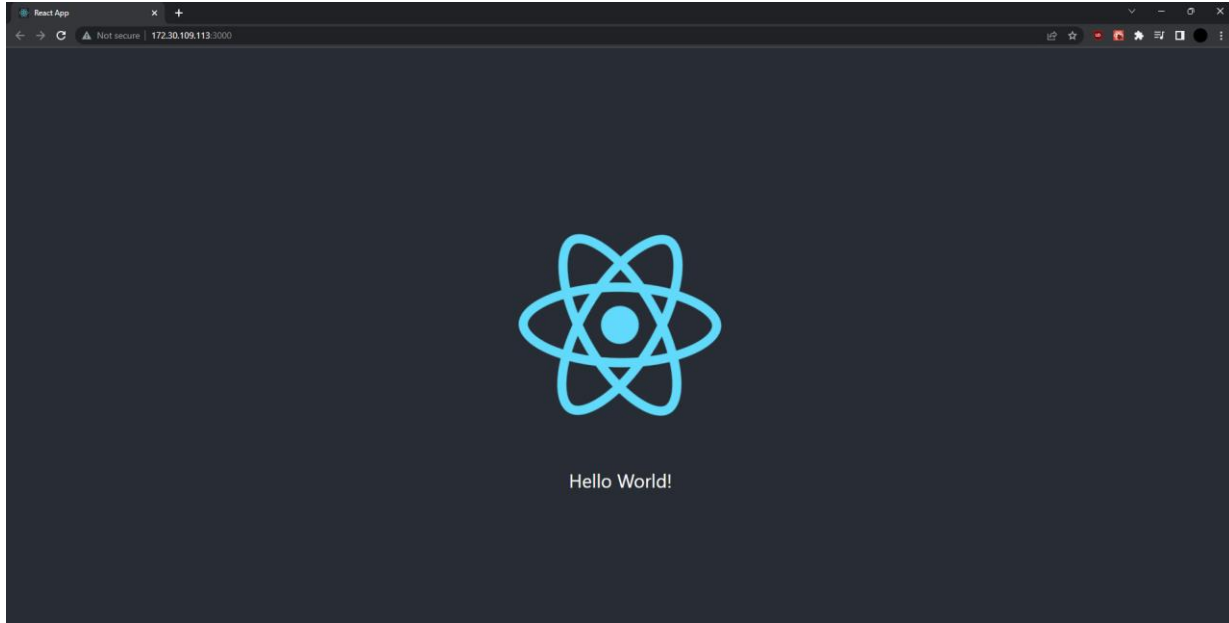
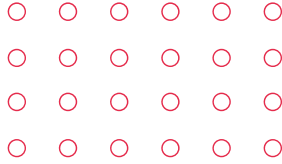


# Create your project

- ❖ `npx create-react-app react-workshop-demo --template typescript`
- ❖ `cd react-workshop-demo`
- ❖ `npm start`

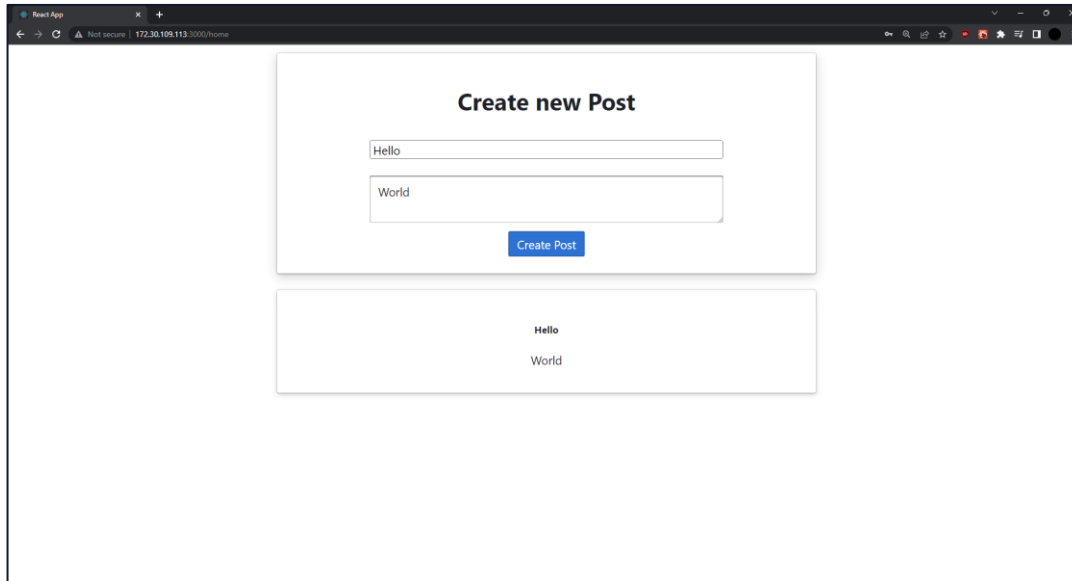


# Hello World!

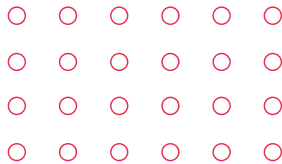


# Home Page

- ❖ A dialog box to create a new post and a card to display a post



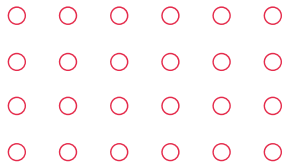
# Installing BlueprintJs



- ❖ Blueprint is a React-based UI toolkit for the web.
- ❖ `npm i @blueprintjs/core`
- ❖ Documentation - <https://blueprintjs.com/docs/>
- ❖ Don't forget to import main CSS files from BlueprintJs packages



# Installing Redux

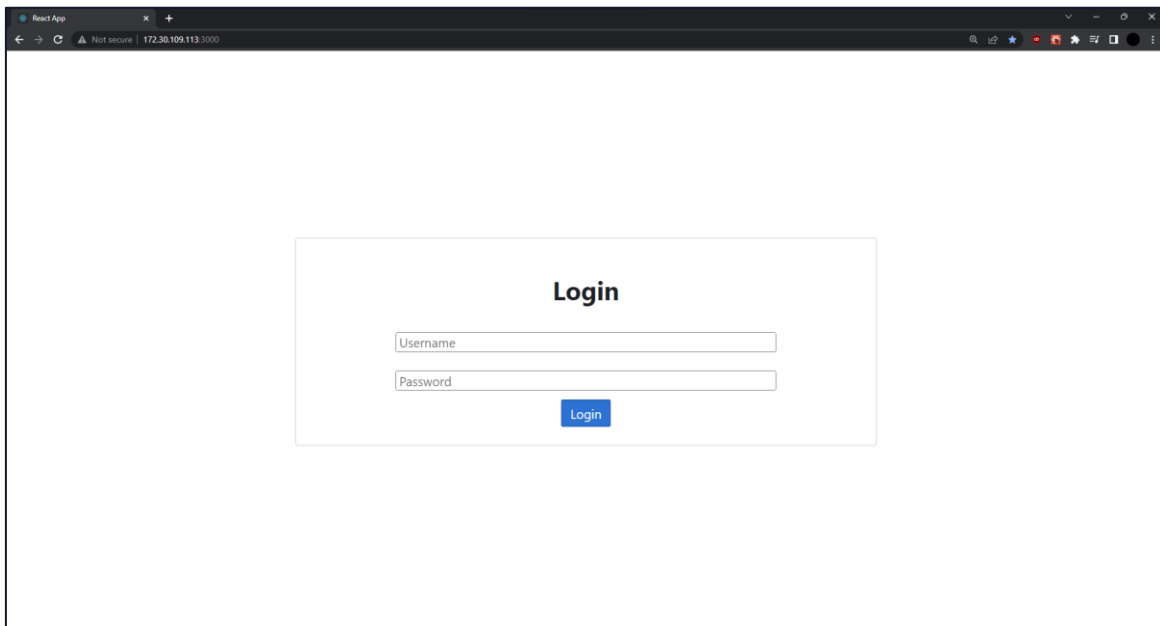


- ❖ `npm install @reduxjs/toolkit react-redux`
- ❖ <https://redux.js.org/introduction/getting-started>
- ❖ <https://redux-toolkit.js.org/introduction/getting-started>
- ❖ Don't forget to wrap the Redux Provider around your application



# Login Page

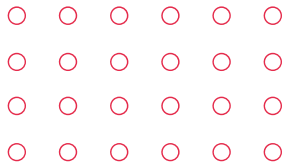
- ❖ A dialog to login and go to home page



A screenshot of a web browser window displaying a login page. The browser's address bar shows "React App" and "172.30.109.113:3000". The login page is centered and contains a form with the title "Login". Below the title are two input fields: "Username" and "Password". A blue "Login" button is positioned below the "Password" field.



# Installing Axios



- ❖ Axios is a promise-based HTTP Client for node.js and the browser.
- ❖ npm install axios
- ❖ <https://axios-http.com/docs/intro>
- ❖ Mock API URL - <https://mocki.io/v1/fb8c09ce-5796-41e1-b0be-ad046f6c87db>



# Installing React Router

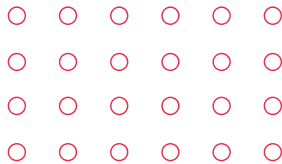


- ❖ `npm install react-router-dom@5.3.4`
- ❖ <https://v5.reactrouter.com/web/guides/quick-start>



# Try it out!

- ❖ Add a log-out button
- ❖ Make Posts editable and deletable
- ❖ Add Comments for Posts
- ❖ Improve on the UI
- ❖ Play around and add your own features!





05

# Source Academy



# Our Tech Stack

## Frontend

Language: TypeScript

Styling: SCSS

Main Library: React

State Management	Redux-Saga
Routing	React Router
Components Management	BlueprintJS
Canvas / Drawing	Konva
Game Rendering	Phaser
Sound Recording	Yareco

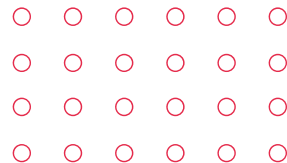
## Backend

Language: Elixir

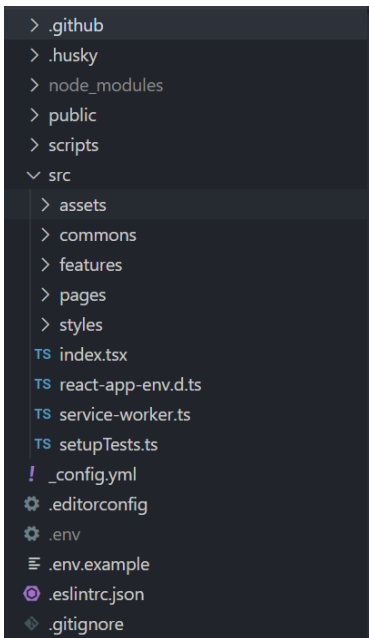
Database: PostgreSQL

Framework: Phoenix

Job Scheduler	Quantum
Authentication	Guardian
DB Abstraction Layer	Ecto



# Frontend Structure



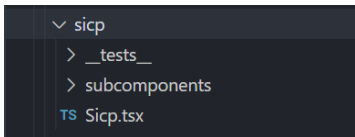
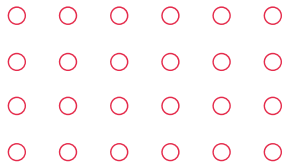
Very **similar** to what we have seen earlier

src/assets	Images, icons
src/styles	SCSS stylesheets to be used by the application
src/commons	<b>Common components</b> found in several pages, eg Editor, REPL, Buttons etc
src/pages	Represents the <b>pages (URL) of the project</b> . Contains the component and containers of the page while <b>subcomponents/ store exclusive components</b> used by the page...
src/features	<b>Contain reducers, actions, utils, sagas, epics/services, helpers</b> . Usually, has <b>corresponding component</b> under pages/ directory.

⋮

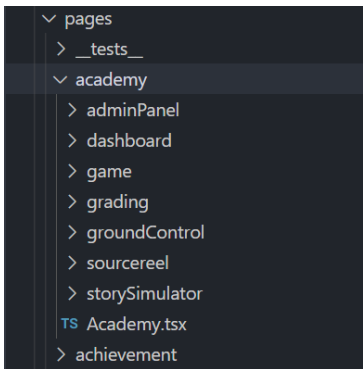
*Disclaimer: Descriptions were mostly taken off the official [Developer's Guide](#)*

# Frontend Structure (Pages)



## Subcomponents

Exclusive components that exist only in that page



## Subpages

Each subpage exists as a separate folder, which can have its own *subcomponents*/folder

E.g., Tabs within a page



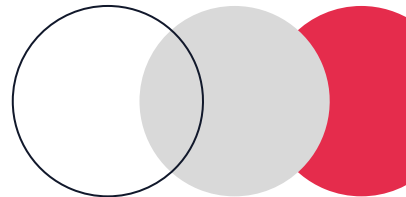
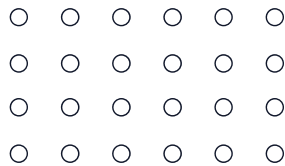


# Solve a bug!

Solve an existing bug in our source  
academy frontend!

[Github #2231](#)





**Thanks for listening!**  
See you around in school!

**Content Material**

*<https://github.com/Junyi00/react-introductory-sa-ay2223>*

