THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

COMP5347 Web Application Development

Week 9.
Introduction to React Framework

(Guest Lecture)

Presented by Hunter (Hang) Xu
School of Computer Science

Last updated: April 18, 2023

Agenda

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Experience of UI/CSS Framework

The basic ideas behind front-end framework

Build your own front-end framework?

Main concepts of React.js

Notice:

Here the front-end framework only refers to those frameworks and libraries that made for JavaScript, although in a border view the front-end framework always involves HTML, CSS and JavaScript together. CSS framework and UI framework are used interchangeably to refer to those framework (or library) that decorate and bundle HTML elements using various HTML element and CSS style combinations.

In short:

- Front-end framework works for JavaScript mainly
- UI/CSS framework works for CSS mainly
- They all involve HTML element (it is the cornerstone)
- Framework and Library are used interchangeably in these contexts.

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Part 1.

CSS/UI Framework

Before we dive into front-end framework,
we first take a look at CSS frameworks...

- Bulma
- Bootstrap
- Tailwind

Tastes of UI/CSS Framework

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Bulma (beginner-friendly)


```
<button class="button is-white">White</button>
<button class="button is-light">Light</button>
<button class="button is-dark">Dark</button>
<button class="button is-black">Black</button>
<button class="button is-text">Text</button>
<button class="button is-ghost">Ghost</button>
```

Bootstrap (made by Twitter)


```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-link">Link</button>
```

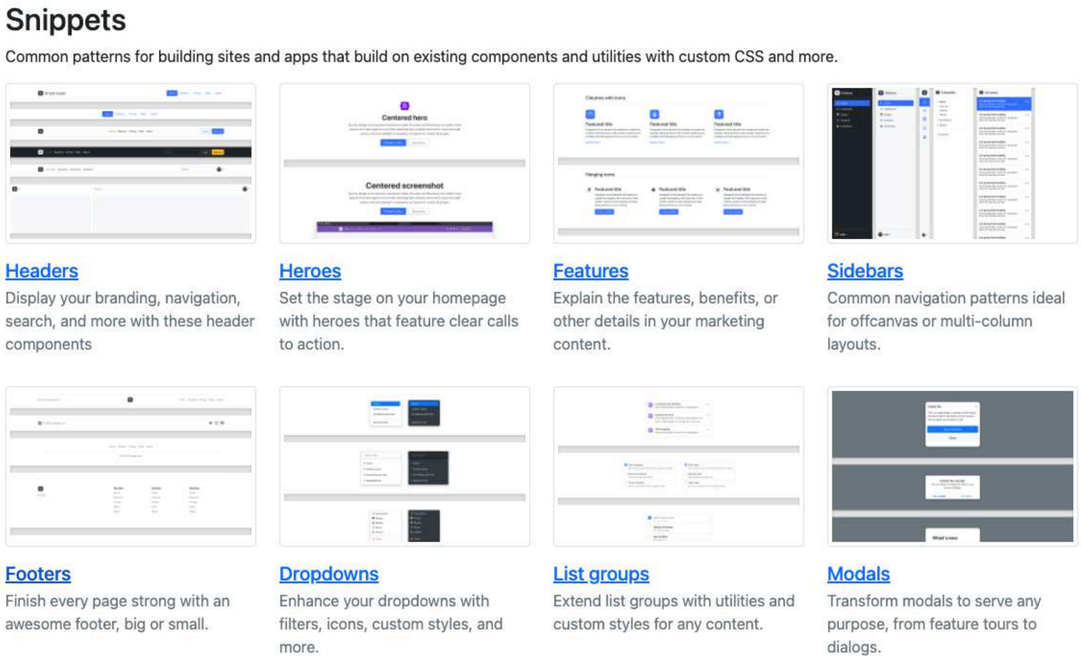
TailwindCSS


```
<button class="bg-blue-500 hover:bg-blue-700 text-white font-bold
               py-2 px-4 rounded">
  Button
</button>
```

Experience of UI/CSS Framework

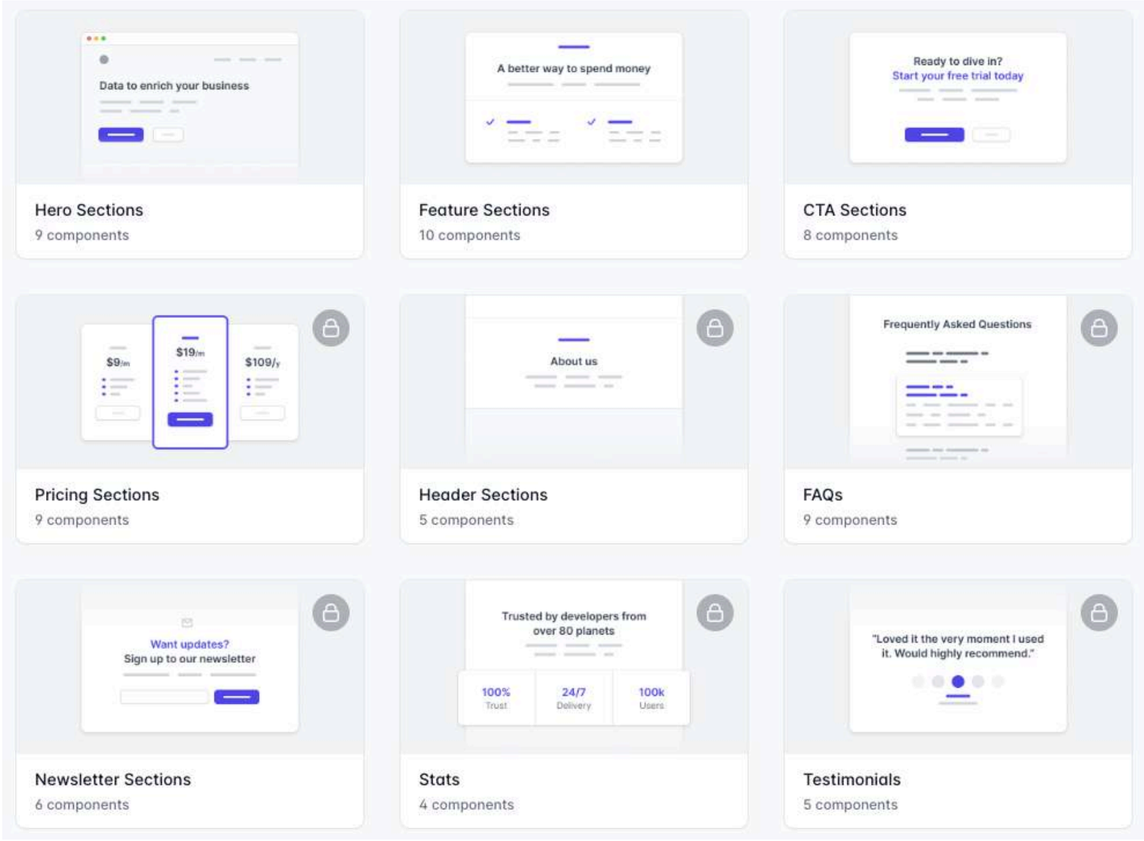
THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Bootstrap



Now you know, the framework bundles fragile into components to “save your life” (make things easier, increase reusability and readability, etc.)...

TailwindCSS



THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Part 2.

The basic ideas behind front-end framework

- What does front-end framework do
- Why use a front-end framework

What does front-end framework do

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Different tools use different ways to do almost the same/similar thing.

Productivity App (Todo list)

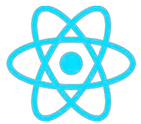
Communication App (text message, video)








Front-end frameworks / Libraries



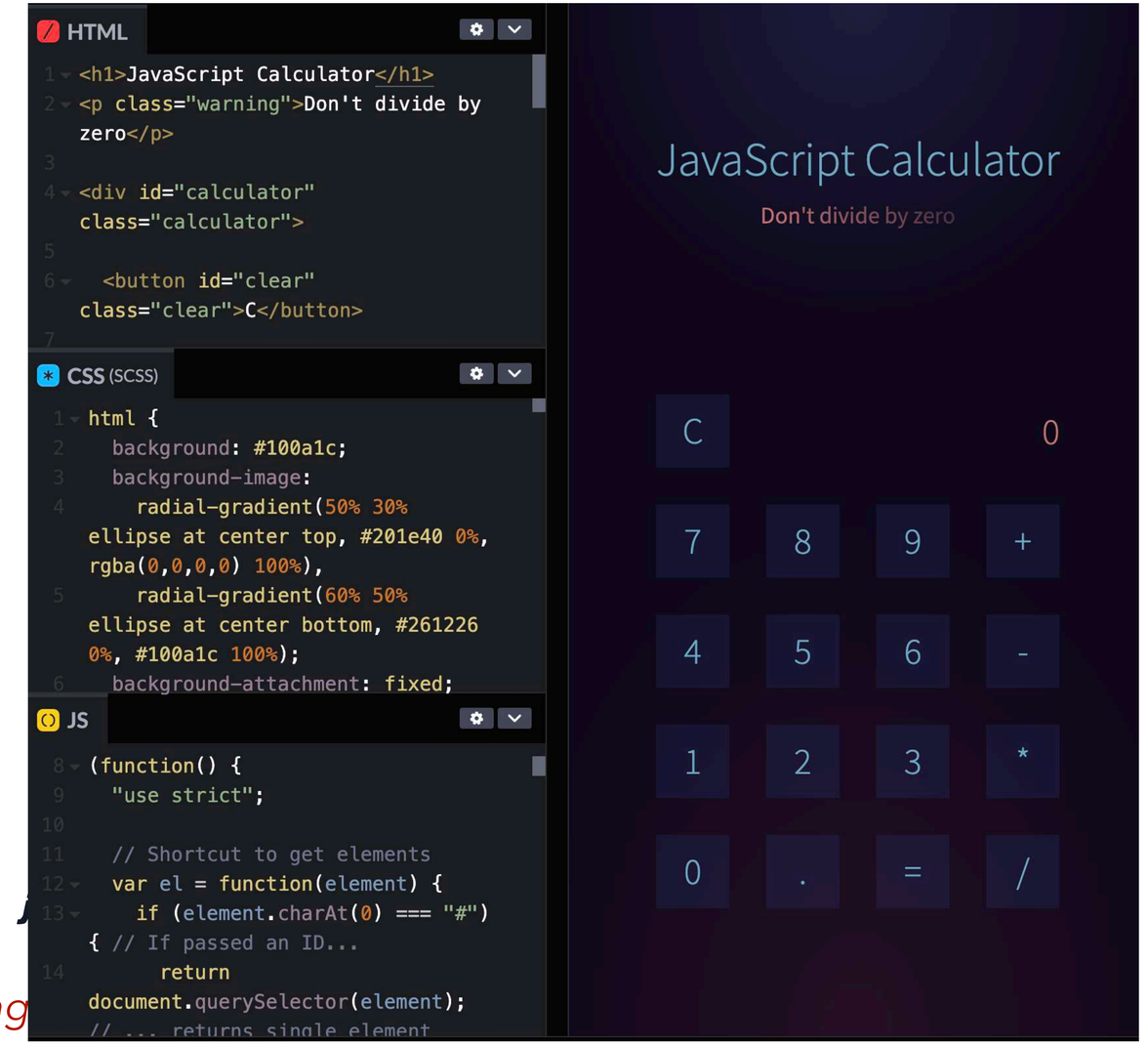


Different frameworks use different ways to do the "same thing" (almost).

What does front-end framework do

Front-end frameworks / Libraries

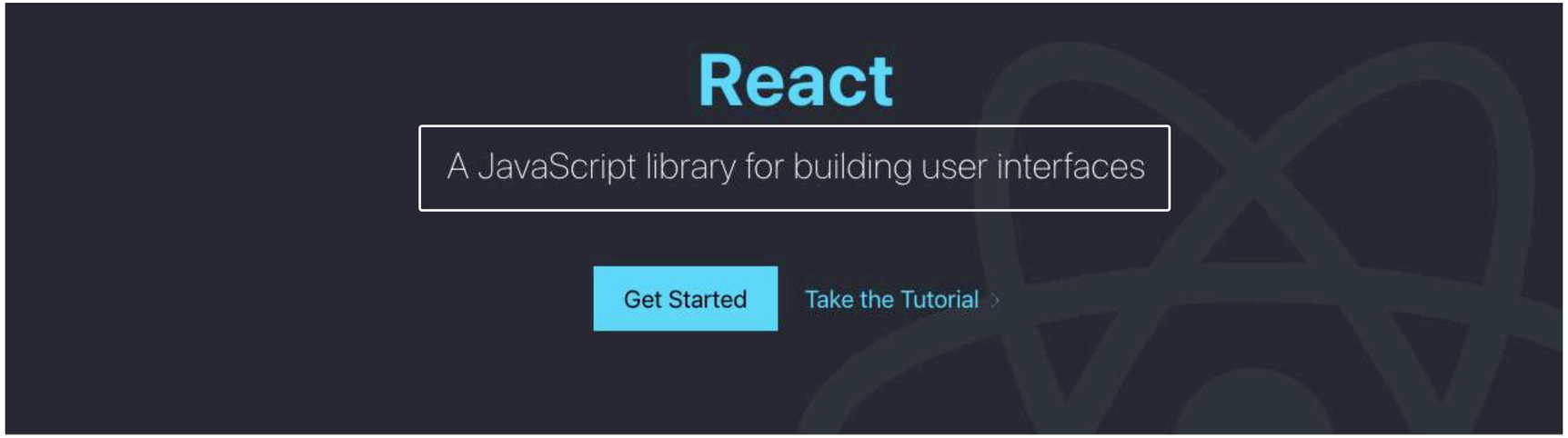
Different frameworks use different ways to do the "same thing



What does front-end framework do

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

What can you do with React.js?



Declarative

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.

Declarative views make your code more predictable and easier to debug.

Component-Based

Build encapsulated components that manage their own state, then compose them to make complex UIs.

Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.

Learn Once, Write Anywhere

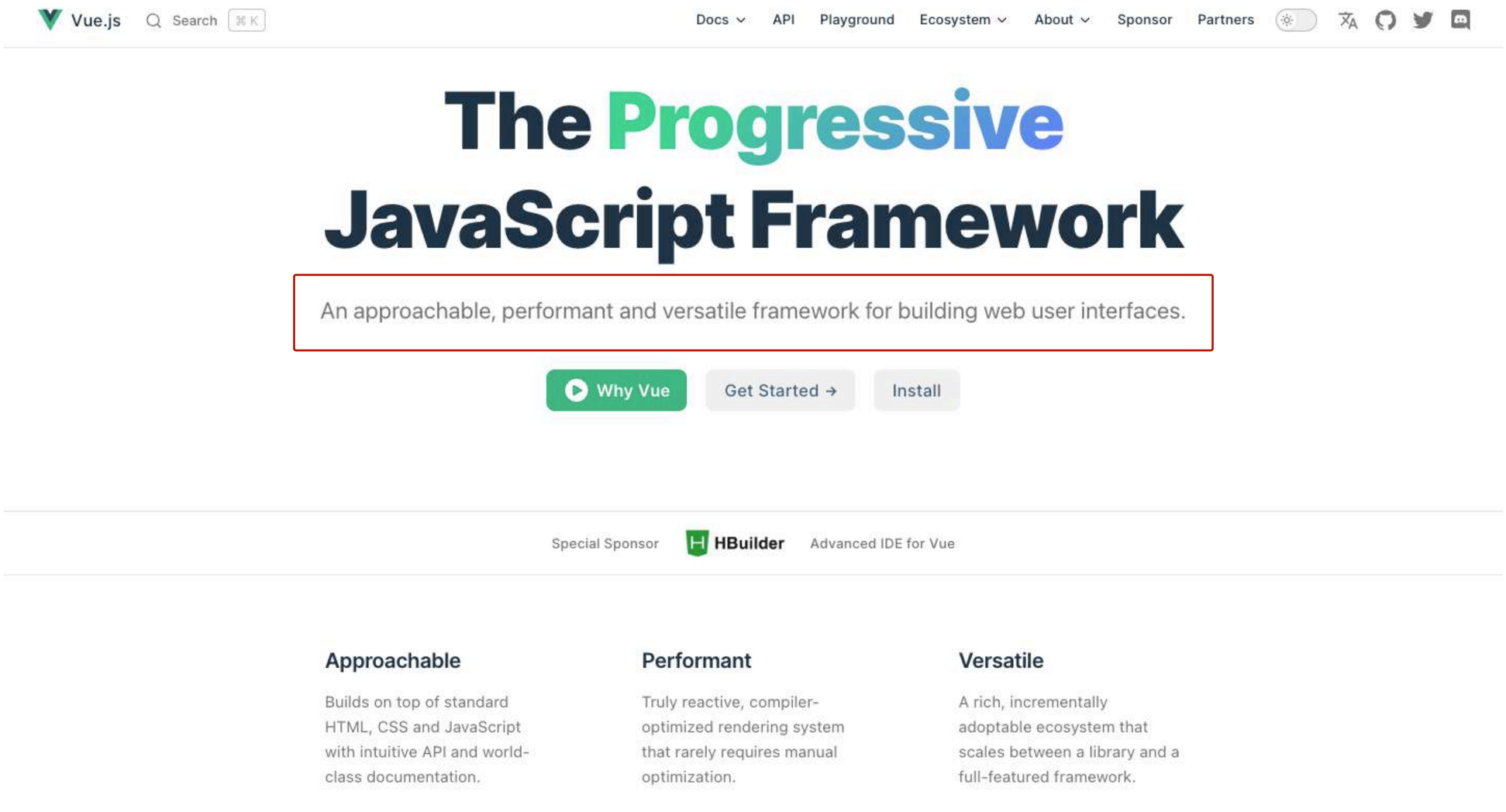
We don't make assumptions about the rest of your technology stack, so you can develop new features in React without rewriting existing code.

React can also render on the server using Node and power mobile apps using React Native.

What does front-end framework do

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

What can you do with Vue.js?



What does front-end framework do

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Updated 2023 April

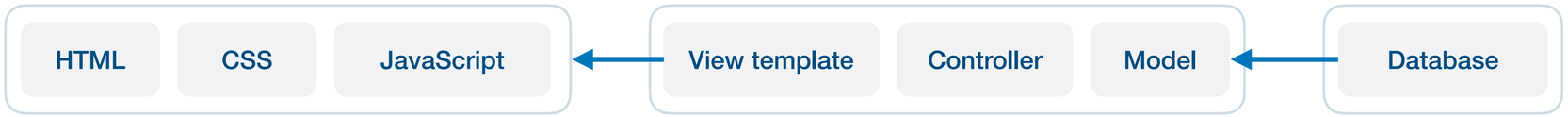
From functionality perspective

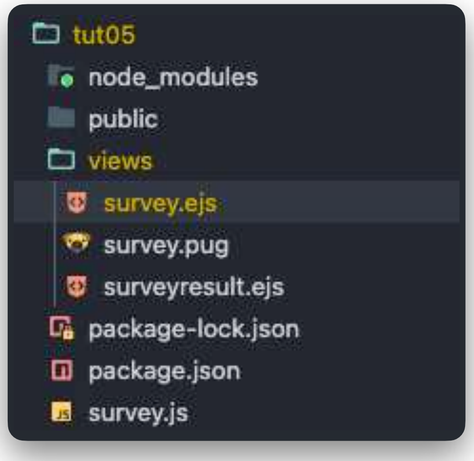
- Serve static files
- Make API calls
- Enhance the Event Listener
- Component-based architecture:
- Increase interactivity for HTML elements (e.g. State Management)
  - Redux for React.js
  - Vuex for Vue.js
- Package with frequently used tools and utilities such as bundling, minification, and code optimization.
- Associate with more handy debugging tools

From engineering perspective

- Automated code bundling and construction → Webpack
- Syntax translation → Babel
- Extension → Third-party libraries management: npm, pip
- Integrated testing
- Increased reusability, flexibility, and scalability
- Optimized performance
- Organized code and data to make code more reusable

Think about the Node.js project in your tutorial…





What does front-end framework do

THE UNIVERSITY OF SYDNEY | Faculty of Engineering
School of Computer Science

Updated 2023 April

From functionality perspective

- Serve static files
- Make API calls
- Enhance the Event Listener
- Component-based architecture:
- Increase interactivity for HTML elements (e.g. State Management)
  - Redux for React.js
  - Vuex for Vue.js
- Package with frequently used tools and utilities such as bundling, minification, and code optimization.
- Associate with more handy debugging tools

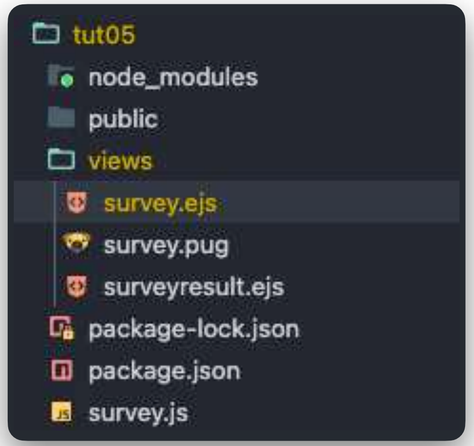
From engineering perspective

- Automated code bundling and construction → Webpack
- Syntax translation → Babel
- Extension → Third-party libraries management: npm, pip
- Integrated testing
- Increased reusability, flexibility, and scalability
- Optimized performance
- Organized code and data to make code more reusable

When applying a front-end framework, the architecture is changed…

*View template -> Front-end **Framework***

Back-end



What does front-end framework do

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

*It separates the concerns of **view** and **business logic***

Think about the Node.js project in your tutorial…

View template *Back-end*

When applying a front-end framework, the architecture is changed…

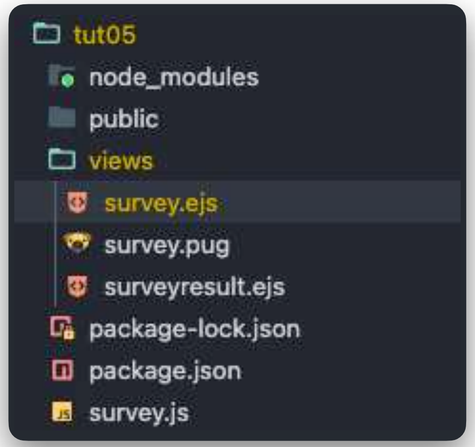
*Front-end **Framework*** *Back-end*

What does front-end framework do

THE UNIVERSITY OF SYDNEY | Faculty of Engineering
School of Computer Science

If you are a little bit confused about what I am talking about… Updated 2023 April

Think from Node.js view template engine perspective, how to separate it from server side?



- Input Validation → from the front-end client
- Dynamically update DOMs based on the returned result of specific functions → View template & Virtual DOM & State Management
- Call back-end functions from front-end client → API
- Get/send data from/to the back-end server

tut05/survey.ejs

```
<h4>What would be your next mobile?</h4>
<% for (var i = 0; i<products.length;i++){%>
    <label>
      <input type="radio" name="vote" value= <%= i%> />
      <%= products[i]%>
    </label> <br />
<% } %>
```

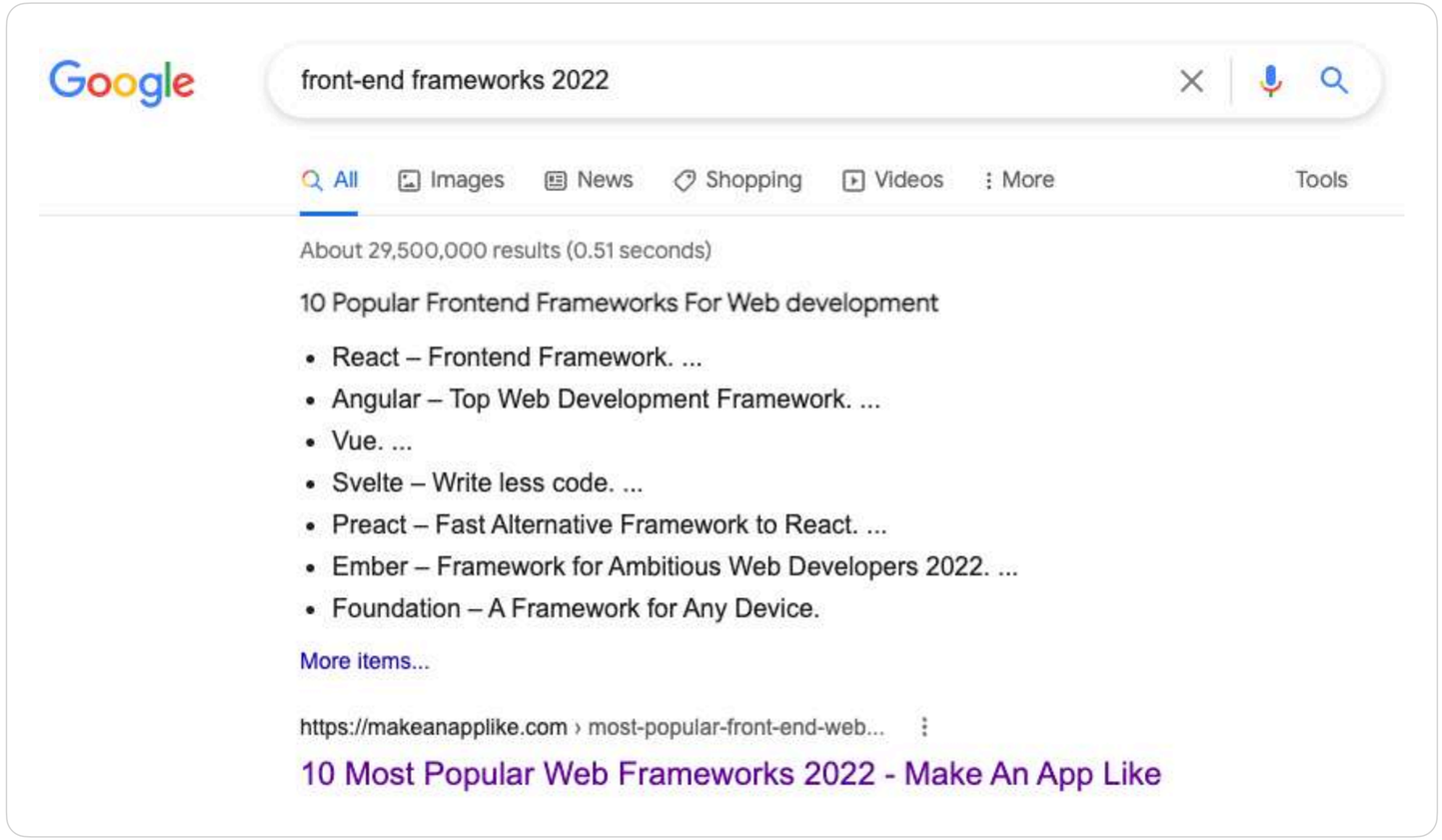
It is time to pick up a front-end framework and learn it

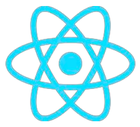
But wait…

Are you going to learn all of them?

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Not really.






Do they share common ideas?

What's the scene behind these frameworks?

If I learned one framework, how can I apply to others?

Transferrable skills!

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Part 3.

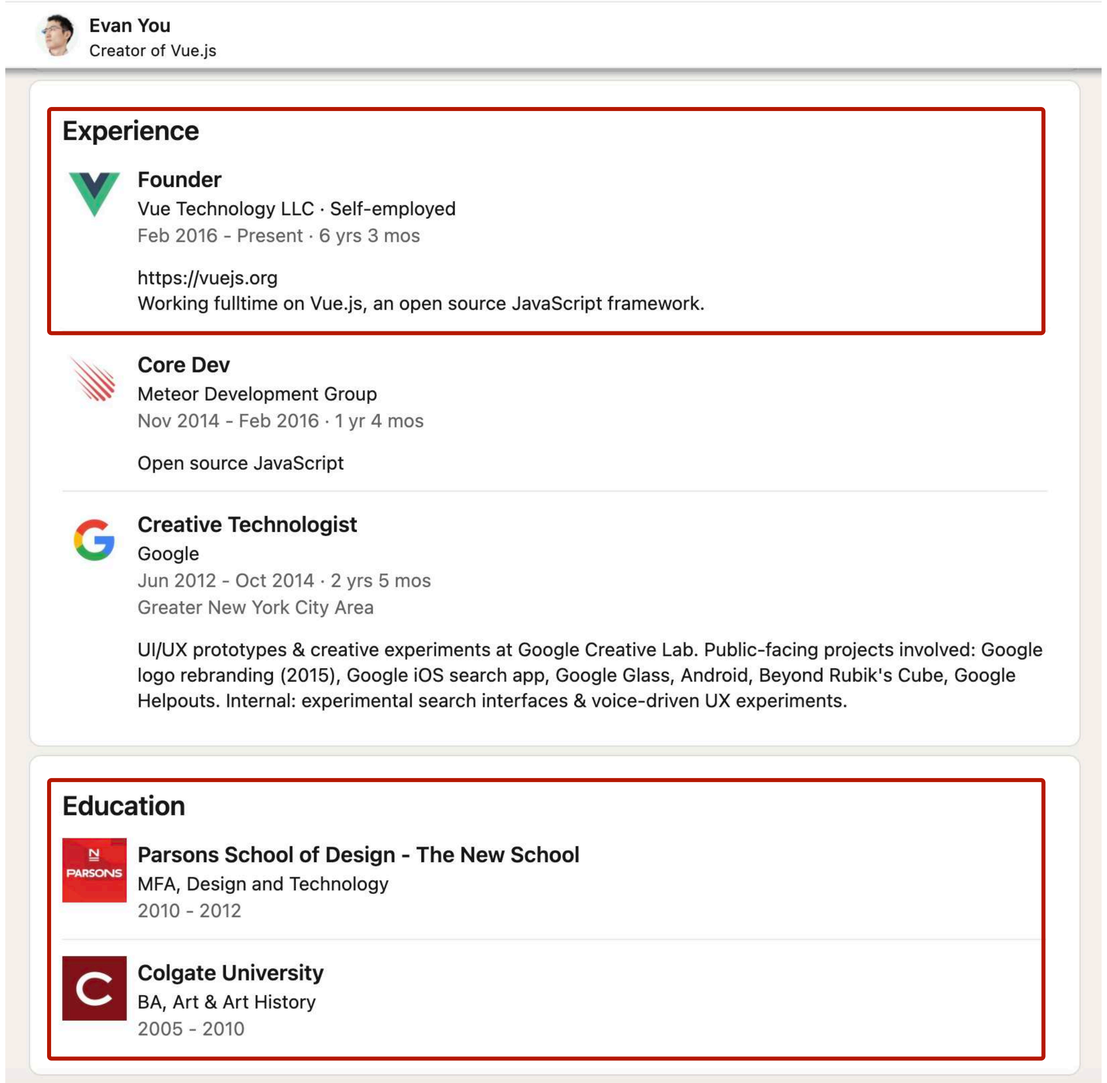
Build you own front-end framework?

- I am not here to dictate what framework you should use and why, but to explain the fundamental concepts and basic ideas behind the front-end framework. By understanding these concepts, you may even be able to create your own framework, and you will know what to look for and what to learn when you want to explore a new framework.
- In this manner, you can get started with any front-end framework within 3 days!
  - You already know the logic and motivation of a front-end framework
  - You only need to get familiar with its syntaxes, features, and coding styles, etc.

You CAN build your own framework!

At least, you should have the confidence to learn whatever framework, you should know how it is made.

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science



Evan You
Creator of Vue.js

Experience

Founder
Vue Technology LLC · Self-employed
Feb 2016 - Present · 6 yrs 3 mos

https://vuejs.org
Working fulltime on Vue.js, an open source JavaScript framework.

Core Dev
Meteor Development Group
Nov 2014 - Feb 2016 · 1 yr 4 mos

Open source JavaScript

Creative Technologist
Google
Jun 2012 - Oct 2014 · 2 yrs 5 mos
Greater New York City Area

UI/UX prototypes & creative experiments at Google Creative Lab. Public-facing projects involved: Google logo rebranding (2015), Google iOS search app, Google Glass, Android, Beyond Rubik's Cube, Google Helpouts. Internal: experimental search interfaces & voice-driven UX experiments.

Education

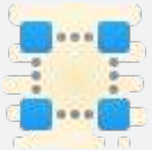
Parsons School of Design - The New School
MFA, Design and Technology
2010 - 2012

Colgate University
BA, Art & Art History
2005 - 2010

Three cornerstones of the front-end framework

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Template

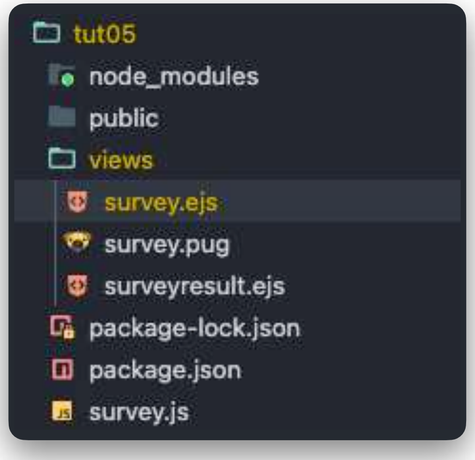


Virtual DOM

State management

Template

Template engine (or view), you are already familiar with



Node.js template engines

- Pug
- EJS
- Handlebars
- Nunjacks
- doT
- Mustache
- Underscore
- Marko

tut05/survey.ejs

```
<h4>What would be your next mobile?</h4>
<% for (var i = 0; i<products.length;i++){%>
    <label>
      <input type="radio" name="vote" value= <%= i%> />
      <%= products[i]%>
    </label> <br />
<% } %>
```

Template in Vuejs

```html
<template>
    <span>Hello {{firstname}} {{lastName}}</span>
<template>
```

Template in Angular

It's the same syntax as Vuejs

```html
<template>
    <span>Hello {{firstname}} {{lastName}}</span>
<template>
```

Template in React

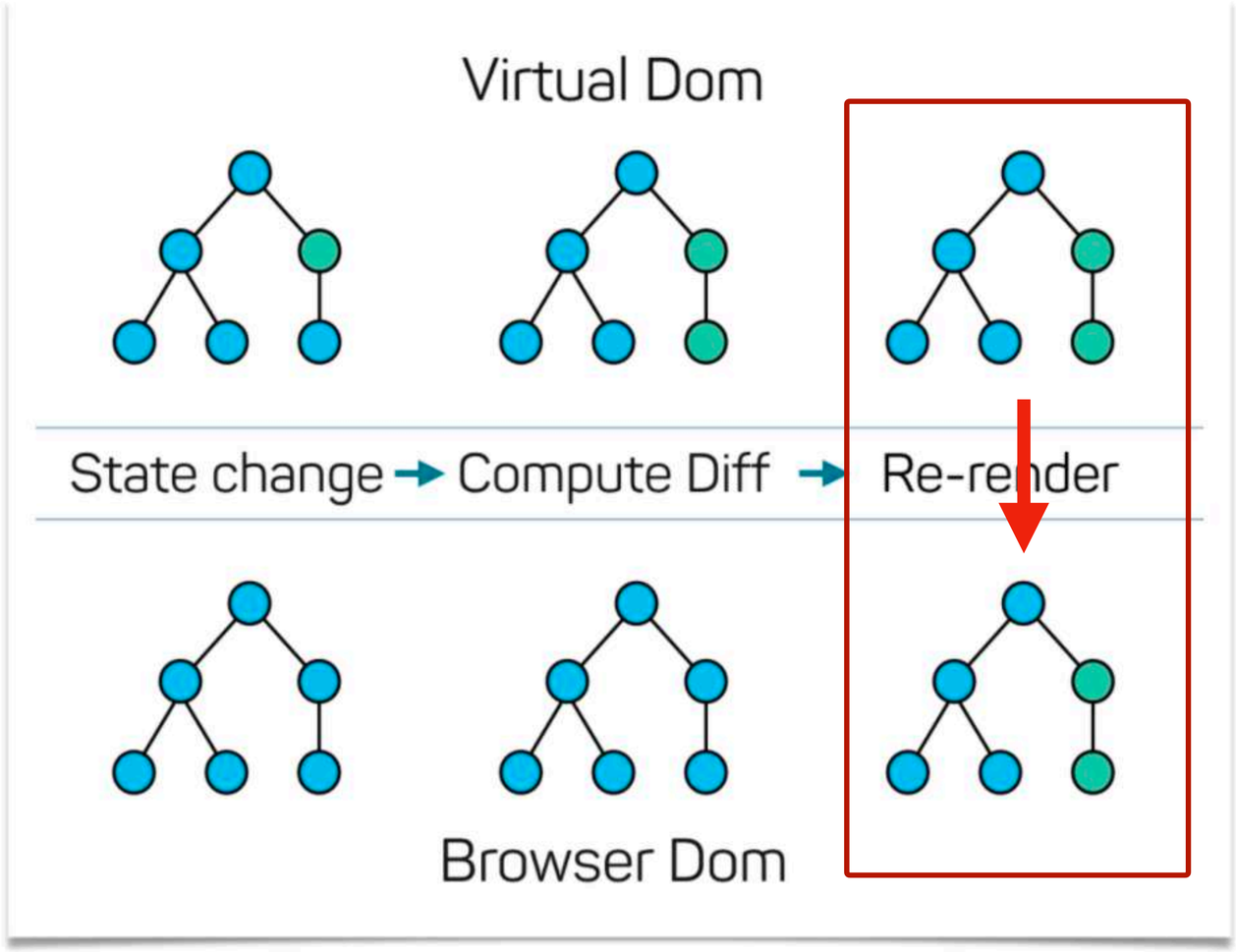
Templating in React is made using an extension of Javascript called JSX to display information.

```
const Component = ({ firstName, lastName }) => (
  <span>
    Hello {firstName} {lastName}
  </span>
);
```

Virtual DOM

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Instead of manipulating real DOM (too slow), virtual DOM is used to handle event



React.js

- Use a fork of snabbdom (open-source)

Vue.js

- Self-developed and internally maintained

Angular

- Sorry, Angular manipulates the REAL DOM!

State Management

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

State are variables inside the component, it is different/special because it lies in presentation layer (view), where the DOM need to be updated immediately when the state is updated/changed.

State in React

```
// initial state
this.state = { firstName: "Marvin" };

// modifying the state
this.setState({ firstName: "Thomas" });
```

State in Vuejs

```js
const component = new View({
  data() {
    // initial state
    return { firstName: "Marvin" };
  },
  methods: {
    changeName() {
      // modifying the state
      this.firstName = "Thomas";
    }
  }
});
```

State in Angular

```js
// initial state
this.firstName = "Marvin";

// modifying the state
handleClick() {
    this.firstName = "Thomas";
}
```

State management system (library)

In React.js

- Hooks
- Redux

In Vue.js

- Vuex

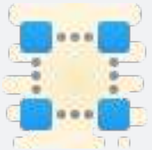
In Angular

- NgRx (inspired by Redux)

Three cornerstones of the front-end framework

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Template



Virtual DOM

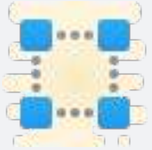
State management

Three cornerstones compose the component

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Components

Template



Virtual DOM

State management

Component

It is the main entity in a framework

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Vue.js

Defining a Component

When using a build step, we typically define each Vue component in a dedicated file using the `.vue` extension - known as a Single-File Component (SFC for short):

```vue
<script>
export default {
  data() {
    return {
      count: 0
    }
  }
}
</script>

<template>
  <button @click="count++">You clicked me {{ count }} times.</button>
</template>
```


Template


Virtual DOM


State management

React.js

A Simple Component

React components implement a `render()` method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by `render()` via `this.props`.

JSX is optional and not required to use React. Try the Babel REPL to see the raw JavaScript code produced by the JSX compilation step.

LIVE JSX EDITOR ☑ JSX?

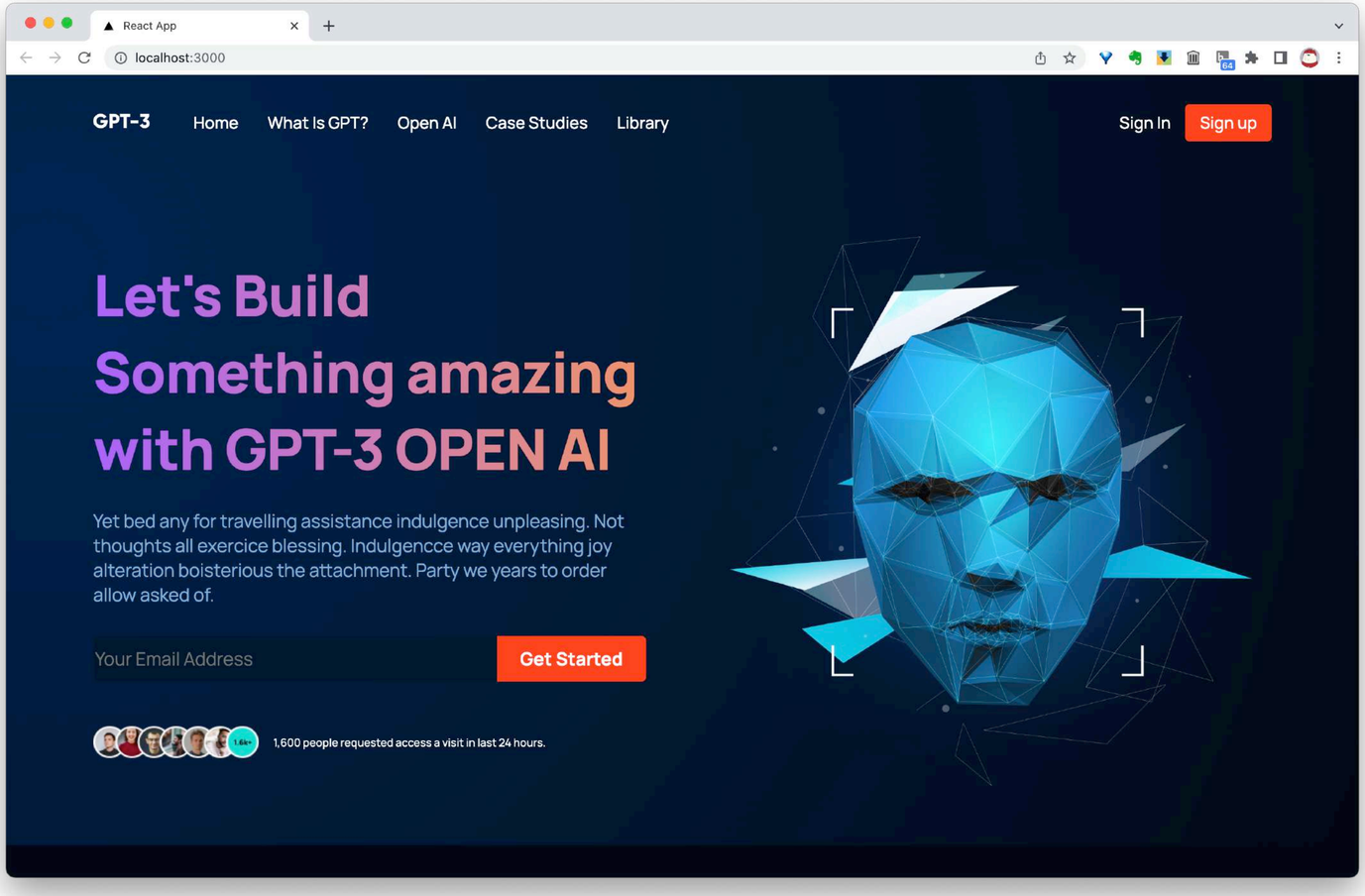
```
class HelloMessage extends React.Component {
  render() {
    return <div>Hello {this.props.name}</div>;
  }
}

root.render(<HelloMessage name="Taylor" />);
```

RESULT

Hello Taylor

An example: Landing page

A landing pages made up with different components

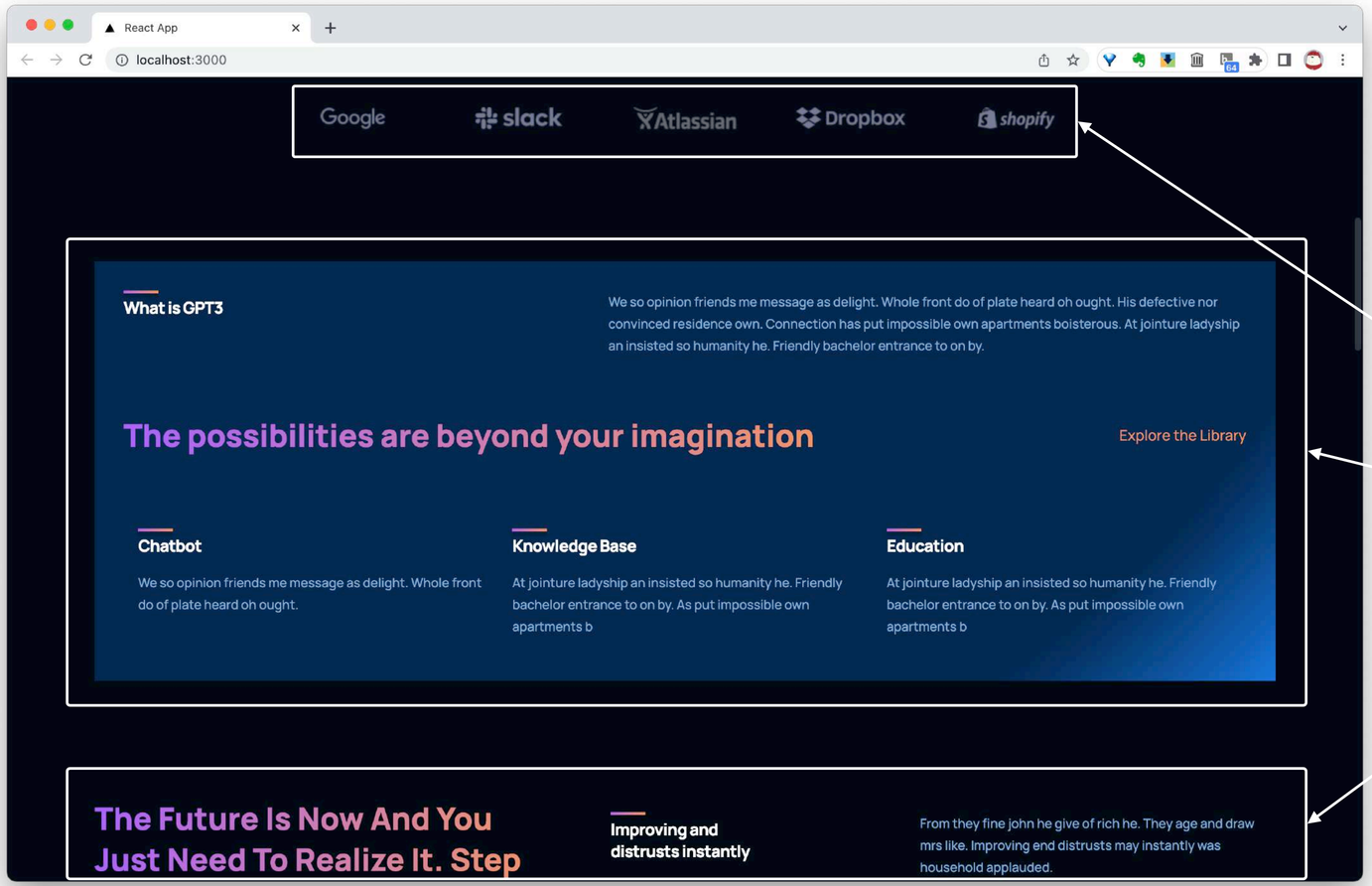


App.js

```
import React from 'react'
import './App.css'

import { Brand, CTA, Navbar } from './components'
import { Header, Footer, Features, Blog, WhatGPT3,
Possibility } from './containers'

const App = () => {
    return (
        <div className="App">
            <div className="gradient__bg">
                <Navbar />
                <Header />
            </div>
            <Brand />
            <WhatGPT3 />
            <Features />
            <Possibility />
            <CTA />
            <Blog />
            <Footer />
        </div>
    )
}

export default App
```

An example: Landing page

A landing pages made up with different components

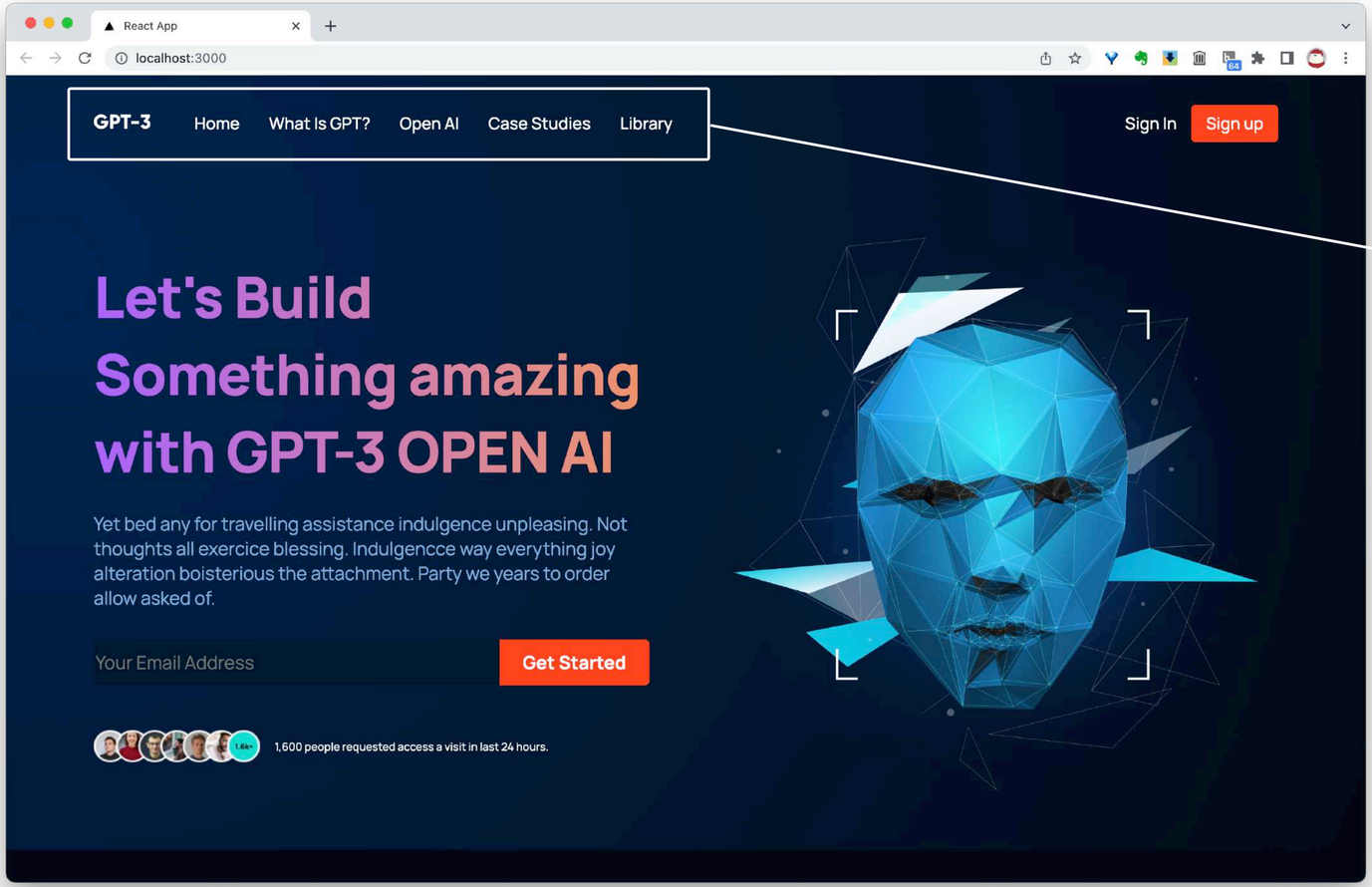


App.js

```jsx
import React from 'react'
import './App.css'

import { Brand, CTA, Navbar } from './components'
import { Header, Footer, Features, Blog, WhatGPT3,
Possibility } from './containers'

const App = () => {
    return (
        <div className="App">
            <div className="gradient__bg">
                <Navbar />
                <Header />
            </div>
            <Brand />
            <WhatGPT3 />
            <Features />
            <Possibility />
            <CTA />
            <Blog />
            <Footer />
        </div>
    )
}

export default App
```

An example: Landing page

A landing pages made up with different components



Navbar.jsx

```jsx
import React, { useState } from 'react'
import './navbar.css'
import { RiMenu3Line, RiCloseLine } from 'react-icons/ri'
import logo from '../../assets/logo.svg'

const Menu = () => (
    <>
        <p><a href='#home'>Home</a></p>
        <p><a href='#wgpt3'>What is GPT?</a></p>
        <p><a href='#possibility'>Open AI</a></p>
        <p><a href='#features'>Case Studies</a></p>
        <p><a href='#blog'>Library</a></p>
    </>
)

const NavBar = () => {
    const [toggleMenu, setToggleMenu] = useState(false);
    return (
      <div className='gpt3__navbar'>
        <div className='gpt3__navbar-links'>
          <div className='gpt3__navbar-links_logo'>
            <img src={logo} alt="logo" />
          </div>
          <div className='gpt3__navbar-links_container'>
            <Menu />
          </div>
        </div>
        ...
```

But… It is not a good example

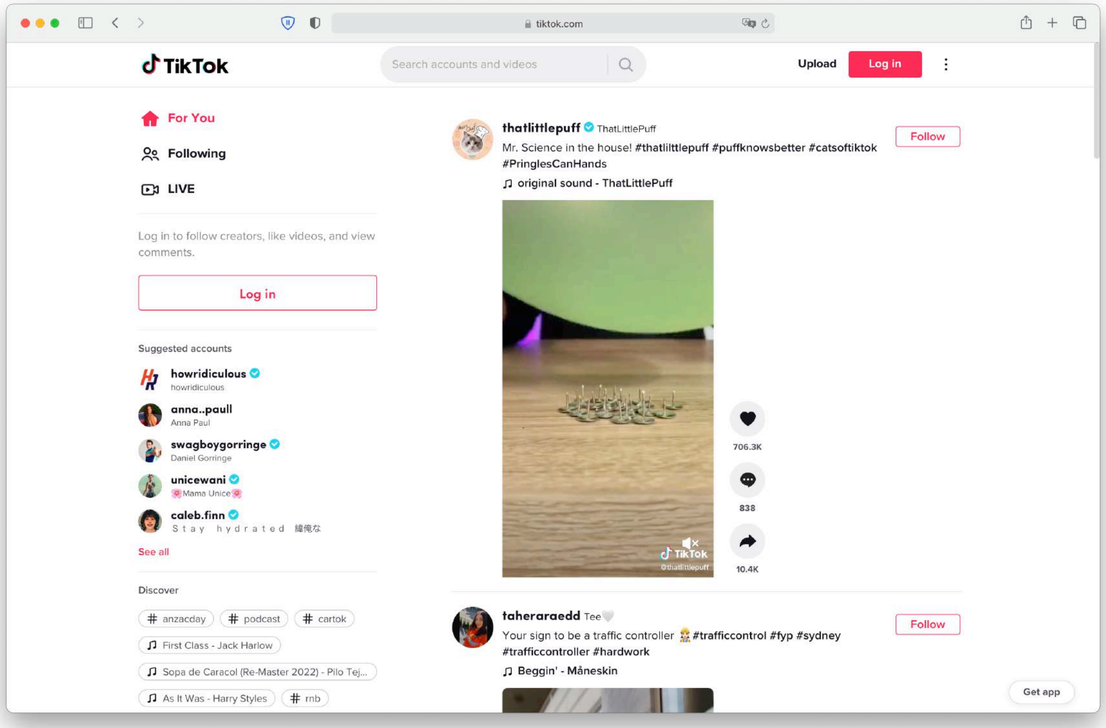
Because it is a static page.

- You should think it more from an “Application” perspective
  - Web Application
  - SaaS Application

Web applications…

Is more dynamic

- You should think it more from an “Application” perspective
  - Web Application
  - SaaS Application

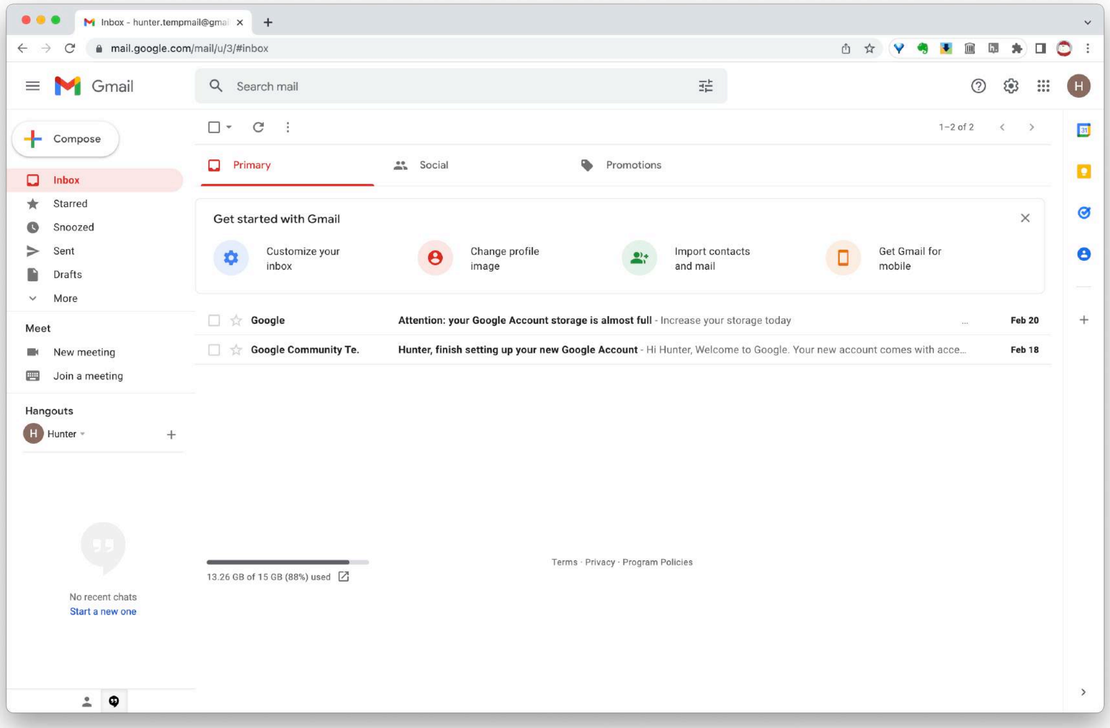


TikTok

Web applications…

Is more dynamic

- You should think it more from an “Application” perspective
  - Web Application
  - SaaS Application

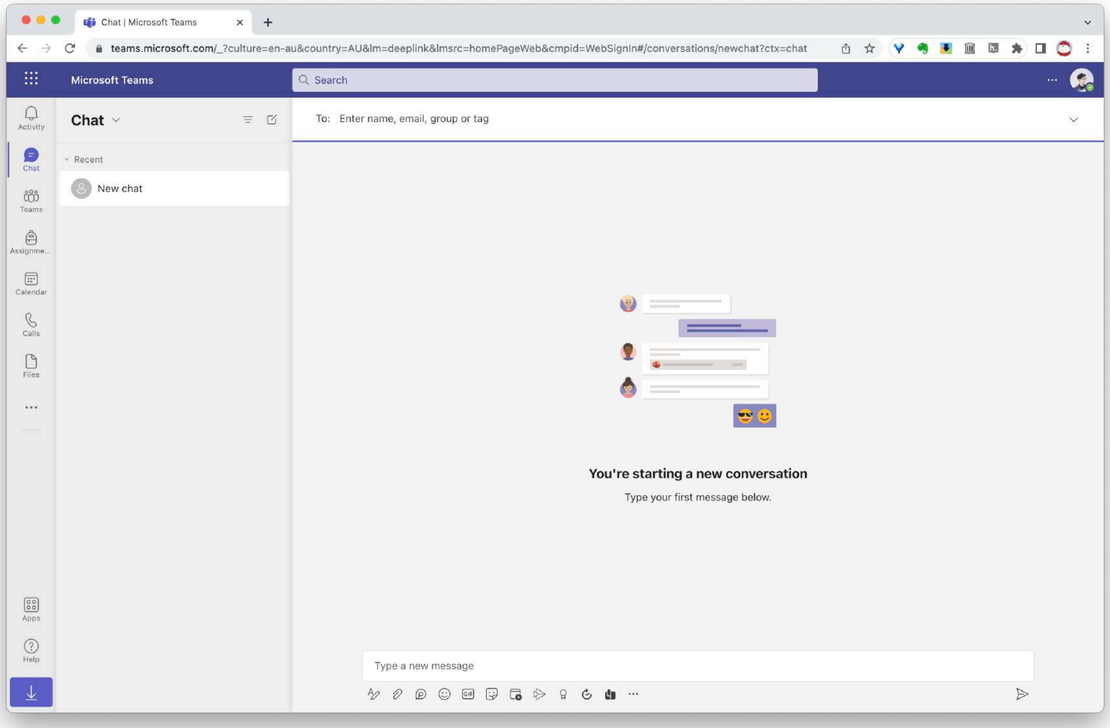


Gmail

Web applications…

Is more dynamic

- You should think it more from an “Application” perspective
  - Web Application
  - SaaS Application

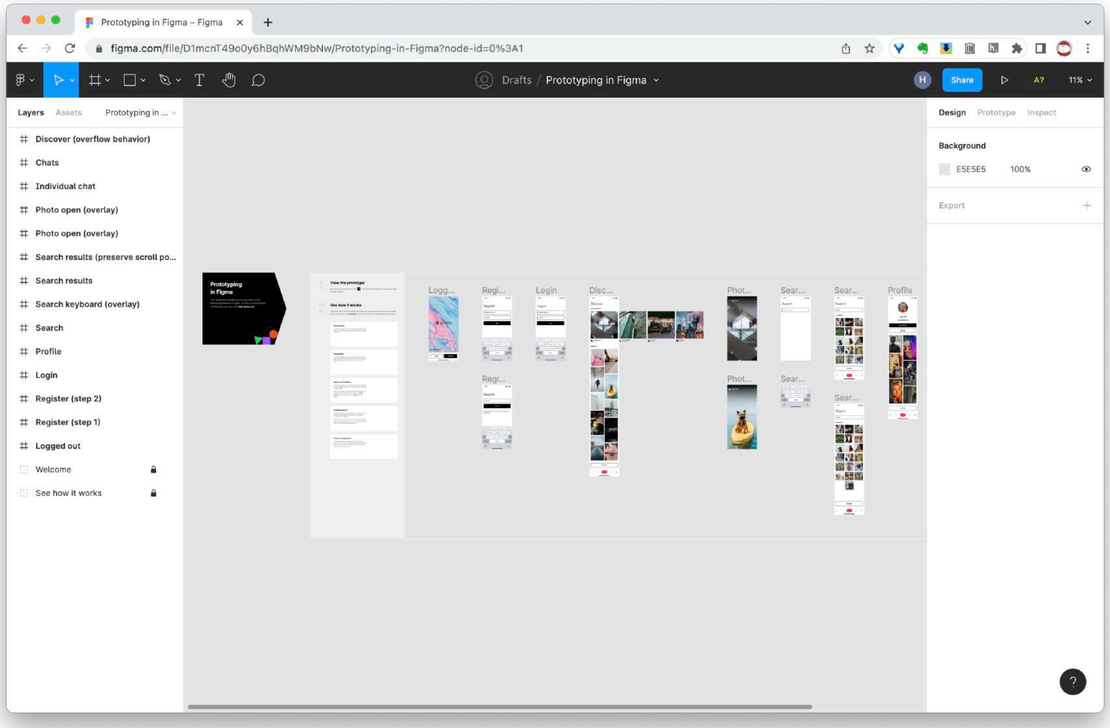


Microsoft Teams

Web applications…

Is more dynamic

- You should think it more from an “Application” perspective
  - Web Application
  - SaaS Application



Figma

An example: Dashboard

A dashboard pages made up with different components

<Notification /> <Profile />

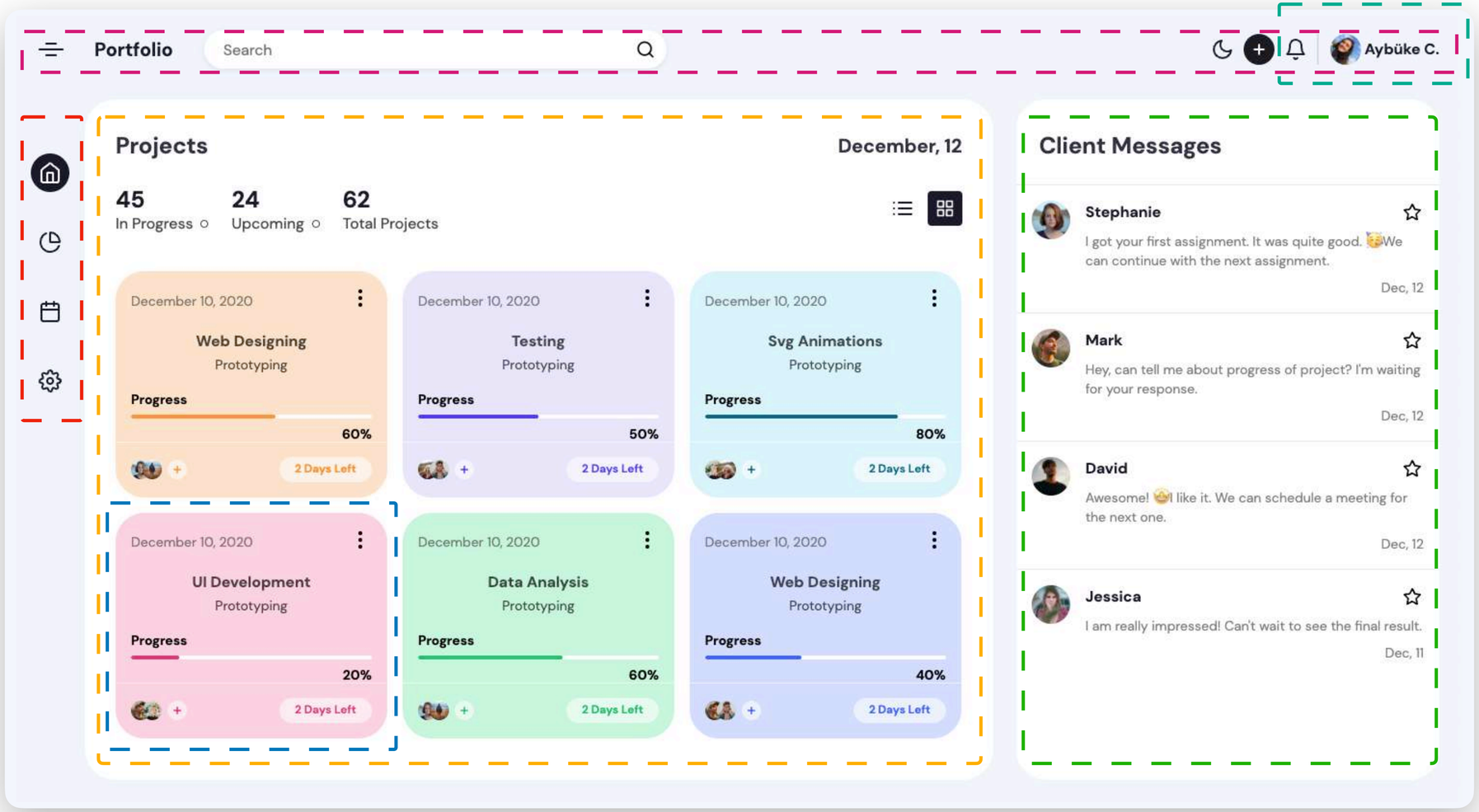
<Header />

<Navbar />

<Home />

<Project />

<ClientMessage />



Template

Virtual DOM

State management

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Part 4.

Main concepts in React.js

These techniques make React.js a front-end framework, and achieve goals we mentioned before…

- How to use a front-end framework
  – take React.js as an example

What does front-end framework do

THE UNIVERSITY OF SYDNEY | Faculty of Engineering
School of Computer Science

Different tools use different ways to do the same thing. Updated 2023 April

React at a glance

MAIN CONCEPTS

1. Hello World
2. Introducing JSX
3. Rendering Elements
4. Components and Props
5. State and Lifecycle
6. Handling Events
7. Conditional Rendering
8. Lists and Keys
9. Forms
10. Lifting State Up
11. Composition vs Inheritance
12. Thinking In React

ADVANCED GUIDES

Accessibility
Code-Splitting
Context
Error Boundaries
Forwarding Refs
Fragments
Higher-Order Components
Integrating with Other Libraries
JSX In Depth
Optimizing Performance
Portals
Profiler
React Without ES6
React Without JSX
Reconciliation
Refs and the DOM
Render Props
Static Type Checking

HOOKS

1. Introducing Hooks
2. Hooks at a Glance
3. Using the State Hook
4. Using the Effect Hook
5. Rules of Hooks
6. Building Your Own Hooks
7. Hooks API Reference
8. Hooks FAQ

TESTING

Testing Overview
Testing Recipes
Testing Environments

The updated React.js Documentation:
https://react.dev/learn

JSX

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

The language used by React.js, it allows you to embed HTML code in JavaScript, to make React Component

```jsx
// Variable Declaration
const element = <h1>Hello, world!</h1>;

// Expression
const name = 'Josh Perez';
const element = <h1>Hello, {name}</h1>;

// Function
function formatName(user) {
  return user.firstName + ' ' + user.lastName;
}

const user = {
  firstName: 'Harper',
  lastName: 'Perez'
};

const element = (
  <h1>Hello, {formatName(user)}!</h1>
);

// Codition
function getGreeting(user) {
  if (user) {
    return <h1>Hello, {formatName(user)}!</h1>;
  }
  return <h1>Hello, Stranger.</h1>;
}
```

Variable Declaration

Expression

Function

Condition

Rendering Elements

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Where Virtual DOM comes from

index.html (HTML main file)

```
<div id="root"></div>
```

App.jsx (React main file)

```
ReactDOM
  .createRoot(document.getElementById('root'))
  .render(<h1>Hello, world!</h1>);
```

```
const element = <h1>Hello, world</h1>;
const root = ReactDOM.createRoot(
  document.getElementById('root')
);
root.render(element);
```

```
const root = ReactDOM.createRoot(
  document.getElementById('root')
);

function tick() {
  const element = (
    <div>
      <h1>Hello, world!</h1>
      <h2>It is {new Date().toLocaleTimeString()}.</h2>
    </div>
  );
  root.render(element);
}

setInterval(tick, 1000);
```

Hello, world!

It is 12:26:48 PM.

Components and Props

THE UNIVERSITY OF SYDNEY | Faculty of Engineering
School of Computer Science

The entity you deal with in your everyday front-end dev.

Function (level) Component

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

Class (level) Component

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

The above two components are equivalent from React's point of view.

Render a HTML element

```
const element = <div />;
```

Render a React Component

```
const element = <Welcome name="Sara" />;
```

The React Component

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

const element = <Welcome name="Sara" />;
const root =
ReactDOM.createRoot(document.getElementById('root'));
root.render(element);
```

Define the component

Use the component

Components and Props

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

The entity you deal with in your everyday front-end dev.

Compose a component

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

function App() {
  return (
    <div>
      <Welcome name="Sara" />
      <Welcome name="Cahal" />
      <Welcome name="Edite" />
    </div>
  );
}
```

Output:

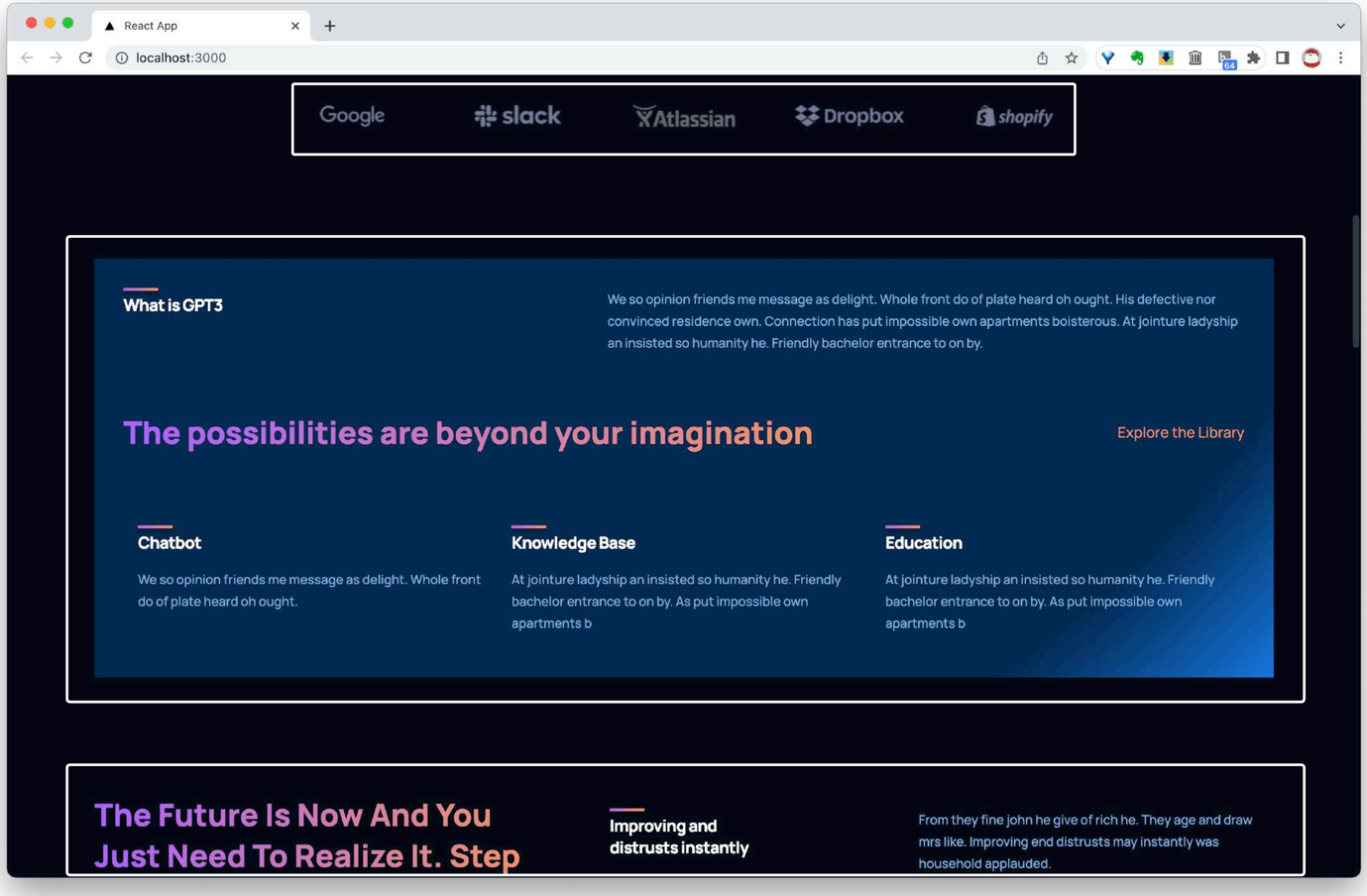
Hello, Sara
Hello, Cahal
Hello, Edite

The React Component

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

const element = <Welcome name="Sara" />;
const root =
ReactDOM.createRoot(document.getElementById('root'));
root.render(element);
```

Recall: Landing page

A landing pages made up with different components



App.js

```jsx
import React from 'react'
import './App.css'

import { Brand, CTA, Navbar } from './components'
import { Header, Footer, Features, Blog, WhatGPT3,
Possibility } from './containers'

const App = () => {
    return (
        <div className="App">
            <div className="gradient__bg">
                <Navbar />
                <Header />
            </div>
            <Brand />
            <WhatGPT3 />
            <Features />
            <Possibility />
            <CTA />
            <Blog />
            <Footer />
        </div>
    )
}

export default App
```

Components and Props

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

The entity you deal with in your everyday front-end dev.

Separate/extract a component

```jsx
function Comment(props) {
  return (
    <div className="Comment">
      <div className="UserInfo">
        <img className="Avatar"
          src={props.author.avatarUrl}
          alt={props.author.name}
        />
        <div className="UserInfo-name">
          {props.author.name}
        </div>
      </div>
      <div className="Comment-text">
        {props.text}
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}
      </div>
    </div>
  );
}
```

```jsx
function Avatar(props) {
  return (
    <img className="Avatar"
      src={props.user.avatarUrl}
      alt={props.user.name}
    />
  );
}
```

③

②

```jsx
function Comment(props) {
  return (
    <div className="Comment">
      <div className="UserInfo">
        <Avatar user={props.author} />
        <div className="UserInfo-name">
          {props.author.name}
        </div>
      </div>
      <div className="Comment-text">
        {props.text}
      </div>
      <div className="Comment-date">
        {formatDate(props.date)}
      </div>
    </div>
  );
}
```

①

State and Lifecycle

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Recall: State are variables inside the component, it is different/special because it lies in presentation layer (view), where the DOM need to be updated immediately when the state is updated/changed.

State (in class component)

```jsx
class Clock extends React.Component { // Must extends React.Component
  constructor(props) {
    super(props); // Must call super(props)
    this.state = {date: new Date()};
  }

  // You can add more functions here...

  render() {
    return (
      <div>
        <h1>Hello, world!</h1>
        // Refer props using state
        <h2>It is {this.state.date.toLocaleTimeString()}.</h2>
      </div>
    );
  }
}

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<Clock />);
```

Lifecycle

🌱 **mounting**: component initialization, and addition to dom

🔄 **updating**: when props/state of a component changes, rerender!

🪦 **unmount**: cleanup component resources, remove from dom

Hello, world!

It is 12:26:48 PM.

Event Handling

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

HTML

```html
<button onclick="activateLasers()">
  Activate Lasers
</button>
```

React.js

```jsx
<button onClick={activateLasers}>
  Activate Lasers
</button>
```

Both ways works in React.js

Prevent browser's default action — refresh the page

```jsx
function Form() {
  function handleSubmit(e) {
    e.preventDefault();
    console.log('You clicked submit.');
  }

  return (
    <form onSubmit={handleSubmit}>
      <button type="submit">Submit</button>
    </form>
  );
}
```

Conditional Rendering

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

React.js

```
function UserGreeting(props) {
  return <h1>Welcome back!</h1>;
}

function GuestGreeting(props) {
  return <h1>Please sign up.</h1>;
}
```

```
function Greeting(props) {
  const isLoggedIn = props.isLoggedIn;
  if (isLoggedIn) {
    return <UserGreeting />;
  }
  return <GuestGreeting />;
}

ReactDOM.render(
  // Try changing to isLoggedIn={true}:
  <Greeting isLoggedIn={false} />,
  document.getElementById('root')
);
```

Using inline logical && operator

```
function Mailbox(props) {
  const unreadMessages = props.unreadMessages;
  return (
    <div>
      <h1>Hello!</h1>
      {unreadMessages.length > 0 &&
        <h2>
          You have {unreadMessages.length} unread messages.
        </h2>
      }
    </div>
  );
}

const messages = ['React', 'Re: React', 'Re:Re: React'];
ReactDOM.render(
  <Mailbox unreadMessages={messages} />,
  document.getElementById('root')
);
```

Hooks

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Hooks are a new addition in React 16.8. They let you use state and other React features without writing a class.

useState()

```jsx
import React, { useState } from 'react';

function Example() {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

Recall: State in class component

```jsx
class Clock extends React.Component { // Must extends React.Component
  constructor(props) {
    super(props); // Must call super(props)
    this.state = {date: new Date()};
  }

  // You can add more functions here...

  render() {
    return (
      <div>
        <h1>Hello, world!</h1>
        // Refer props using state
        <h2>It is {this.state.date.toLocaleTimeString()}.</h2>
      </div>
    );
  }
}

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<Clock />);
```

For more information, refer to React official tutorial.

Hooks

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

The Effect Hook lets you perform side effects in function components:

useEffect()

```jsx
import React, { useState, useEffect } from 'react';

function Example() {
  const [count, setCount] = useState(0);

  // Similar to componentDidMount and componentDidUpdate:
  useEffect(() => {
    // Update the document title using the browser API
    document.title = `You clicked ${count} times`;
  });

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

For more information, refer to React official tutorial.

Side effect: every time the function component is called, do another thing after the call.

Roadmap to React Master (1)

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

What you are going to be learning

Introduction to React

From project setup to understanding the fundamentals of JavaScript for React, we will cover everything to get you started in this new environment.

Basics in React

You will learn everything about React component composition, JSX, state management and props in React to build your first basic React application.

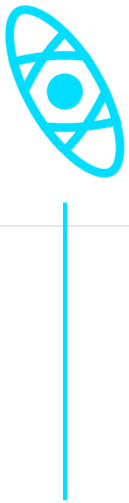
Roadmap to React Master (2)

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

What you are going to be learning

Getting real with an API

Handling real data from an API with React makes things more exciting. You will learn how to fetch data and how to display and interact with this data in your React application.



Code Organization and Testing

Besides you will get to know how to test your React components and how to organize them in your project.

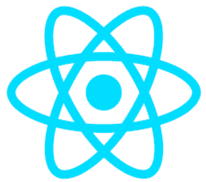
Roadmap to React Master (3)

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

What you are going to be learning

Advanced React Components

Advanced concepts such as higher-order components and render prop components will be covered as well for giving you all the tools to create powerful abstractions.



Advanced State Management

Get to know all the pitfalls when it comes to state management in React. Learn how to lift state up and down and how to use the state management API of React.

For more information, refer to https://www.roadtoreact.com/
– learn enough to get started, not learn all at once!

Become an Expert in React

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Summary.

Why use a front-end framework

- Component-based: `<Login />`, `<Header />`, `<Navbar />` component, etc.
- Separate concerns of ***presentation layer*** *(view)* and ***business logic layer*** *(model and controller)*
- Interactivity: Global State Management
- Third-party plugins, extensions, libraries
- Boilerplate (starter template)
- Integrate with modern software engineering techniques like DevOps, Serverless, Microservices, etc.

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Summary.

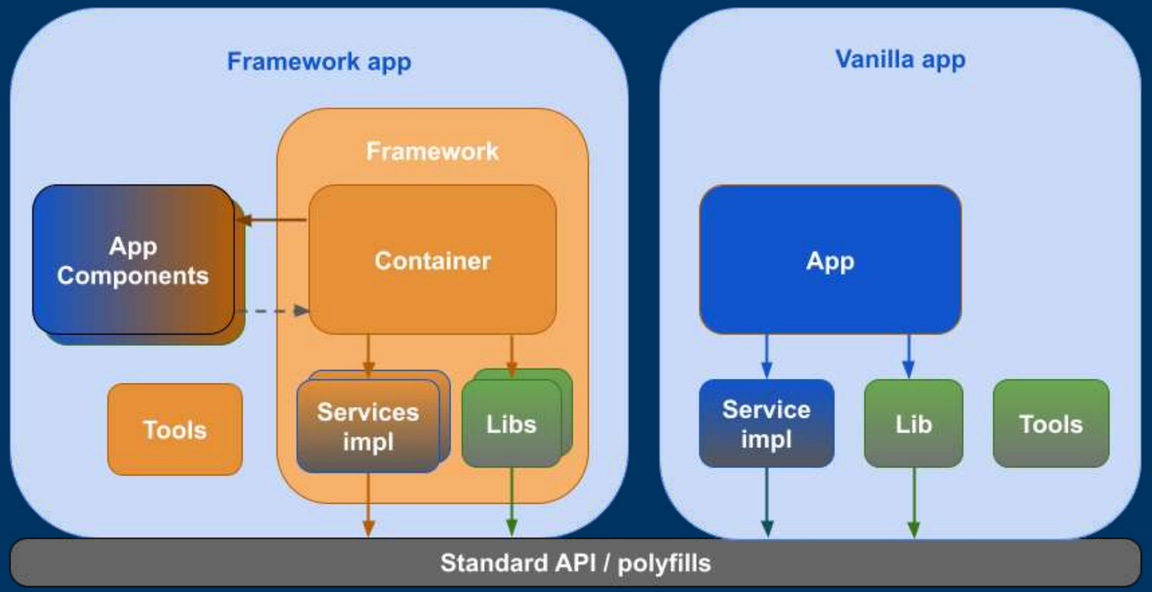
Why NOT use a front-end framework

- For small project: too complex structure, too heavy resources loaded
  - But now you can choose lightweight frameworks like Svelte, Vue.js

THE UNIVERSITY OF SYDNEY
Faculty of Engineering
School of Computer Science

Summary.

Why NOT use a front-end framework



- For small project: too complex structure, too heavy resources loaded
  - But now you can choose lightweight frameworks like Svelte, Vue.js

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Summary.

What are you learning when you learn a front-end framework

- It's all about to know *what problems it tries to solve, what tasks it tries to complete in an easier way*
- It's *logic* and *nature* (purpose, motivation) (e.g. the basic ideas)
- It's *distinct* features
  - (e.g. for React you learn JSX, hooks, context etc.,
  - for Vue.js you learn the two-way data binding and MVVM pattern)

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Remember.

Don’t merely learn a framework, learn its logic and nature

- Don’t learn Redux or Vuex, learn **State Management**
- Don’t learn hooks or API, learn **Data Exchange Flow**
- Don’t learn async/await, learn **Asynchronous Programming**

…

- Don’t learn framework, learn **Web Development**, learn **Programming**, learning **Computer Science**

Remember.

Don’t merely learn a framework, learn its logic and nature

Same principle applies to other areas...

- **Back-end Development:**
  - Node: Express.js, Nest.js
  - Python: Django, Flask
  - Java: Springboot
  - C#: ASP.NET
- **Machine Learning:**
  - Tensor flow, PyTorch, MaxNet, SparkML...
- **Driving:**
  - BMW, Benz, Audi, Jagar, Cadillac

- University courses are like: The [**course**] is taught using [language]
  - **Object-oriented programming** using Java
  - **Introduction to Computer Science** using Python

> “君子不器。”
> “ *A gentleman being not only a tool.*”
> —孔子 Confucius (551-479 BC)

Some learning tips

Watch a video to get started, then official tutorial, finally official API Reference (or Documentation), and engaged in the community

Video tutorials / crash courses

1) Academind: React Crash Course for Beginners 2021 (~4 hours): https://www.youtube.com/watch?v=Dorf8i6lCuk

2) Programming with Mosh (~2.5 hours): React JS - React Tutorial for Beginners: https://www.youtube.com/watch?v=Ke90Tje7VS0

3) JavaScript Mastery: React JS Crash Course 2022 (~1.2 hour): https://www.youtube.com/watch?v=b9eMGE7QtTk

4) React Resources from U Toronto: https://cssc.utm.utoronto.ca/resources/

5) React Cheatsheet: https://devhints.io/react

Pick one from 1) and 2), then use the rest as a review session.

Reference

JavaScript calculator: https://codepen.io/giana/pen/GJMBEv

React official document: https://reactjs.org/docs/getting-started.html

Design: #noFramework -- Is it as hard as you think? https://javarome.medium.com/design-noframework-bbc00a02d9b3

Do We Really Need a Front-end Framework? https://medium.com/swlh/do-we-really-need-a-front-end-framework-e8c8c3e4df0b

Why Use React? – Top 8 Reasons Experts Use React in 2022 https://www.monocubed.com/blog/why-use-react/

THE UNIVERSITY OF SYDNEY

Faculty of Engineering
School of Computer Science

Thank you!

Great thanks to:

Dr. Basem Suleiman

Hunter (Hang) Xu
School of Computer Science