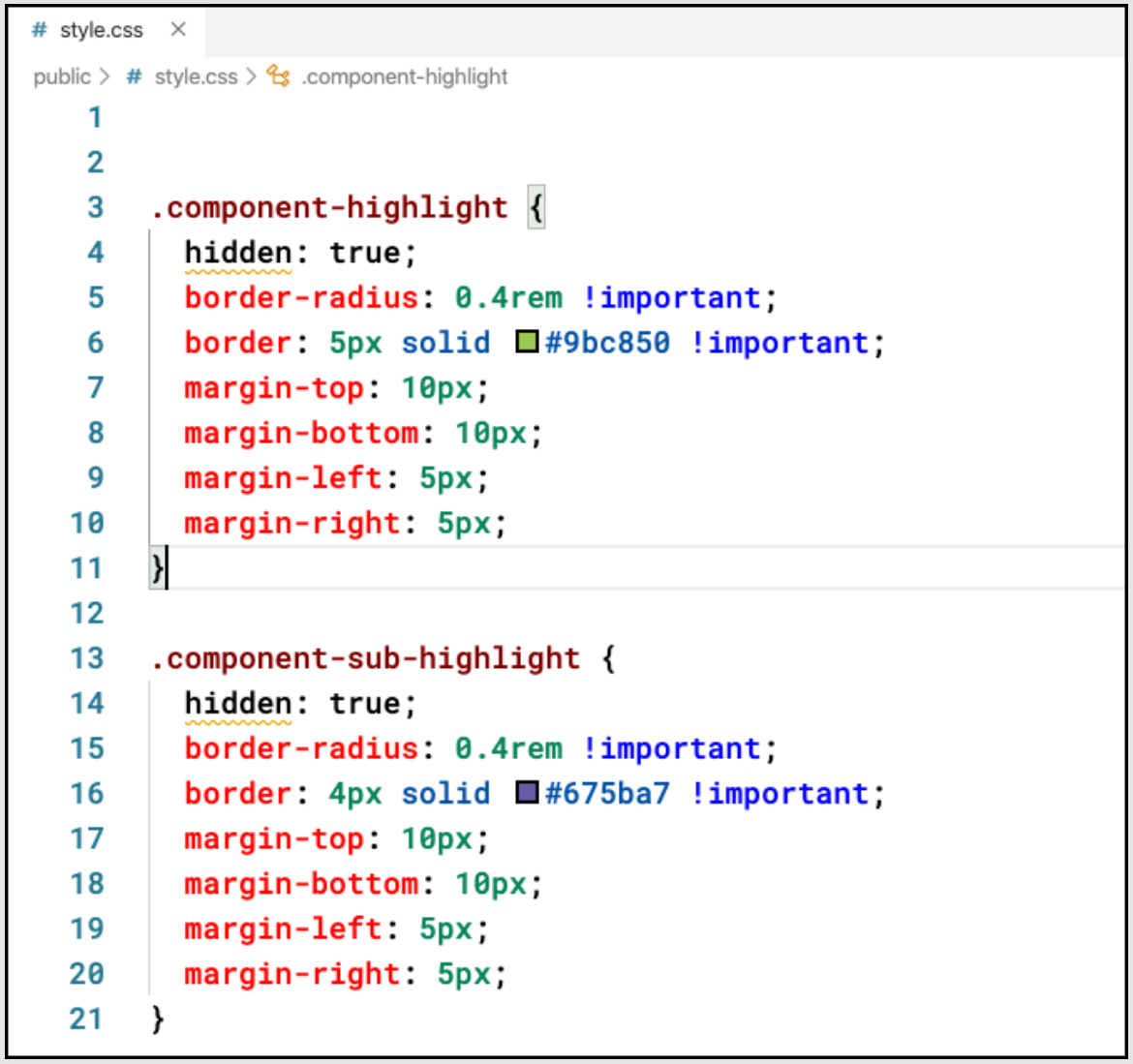
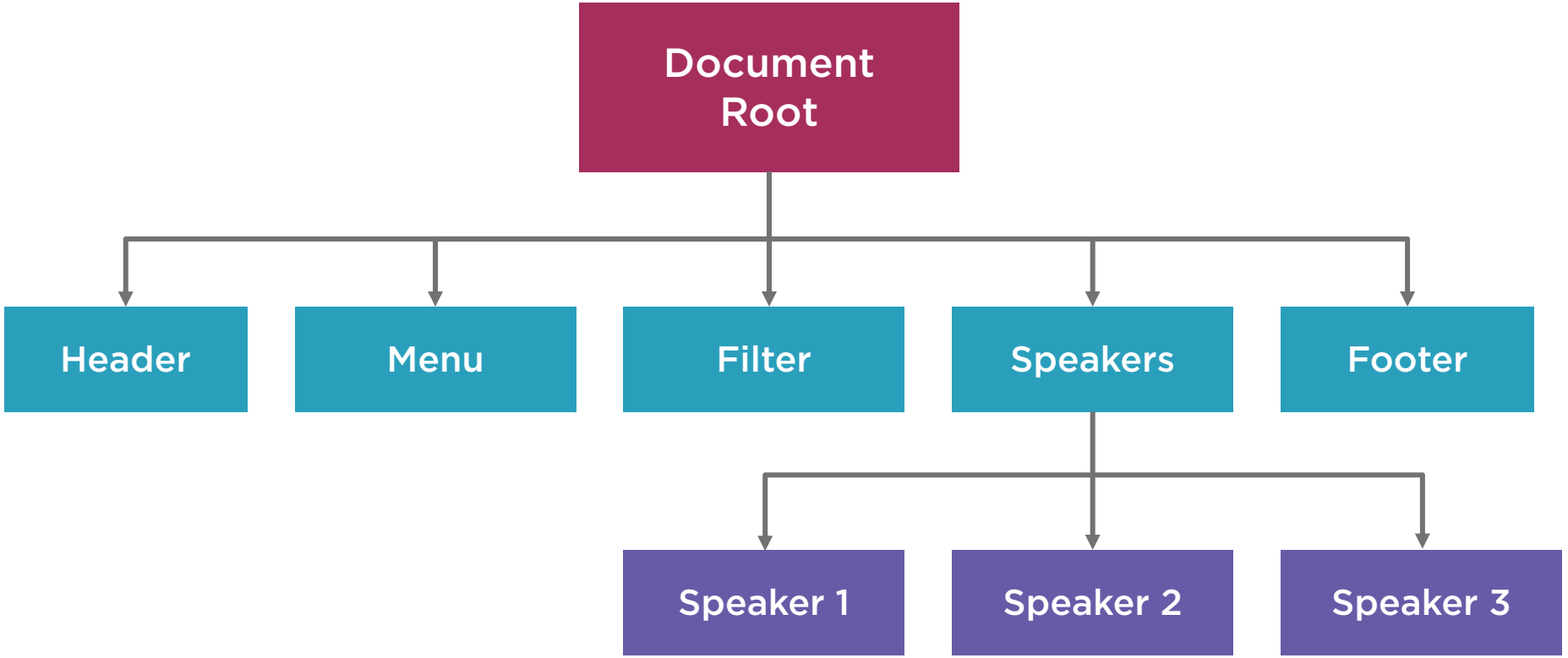
Building a React Web App and Refactoring into Components

Peter Kellner
DEVELOPER, CONSULTANT AND AUTHOR

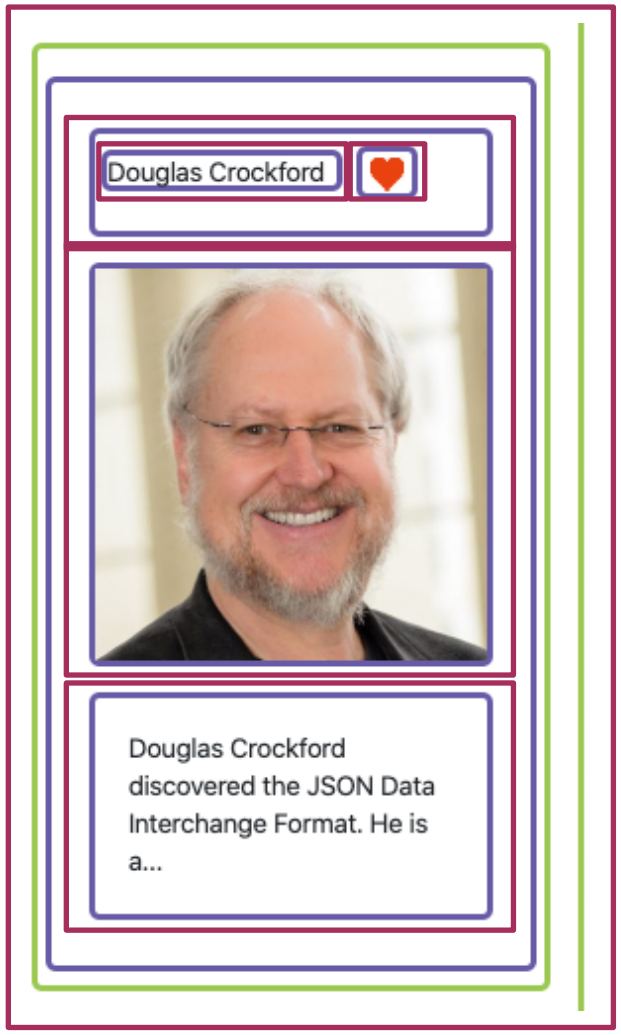
@pkellner linkedin.com/in/peterkellner99 PeterKellner.net

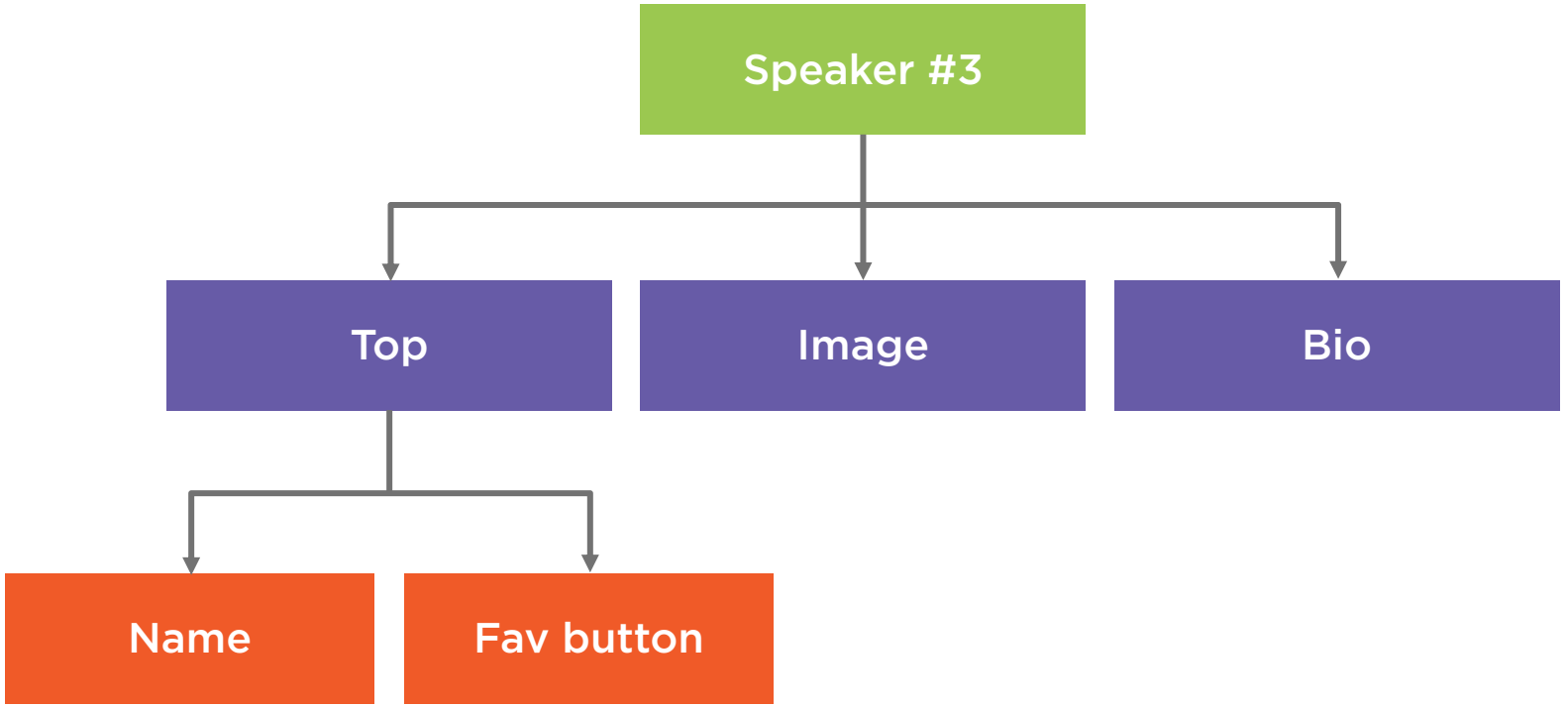
CSS Style Example (style.css)

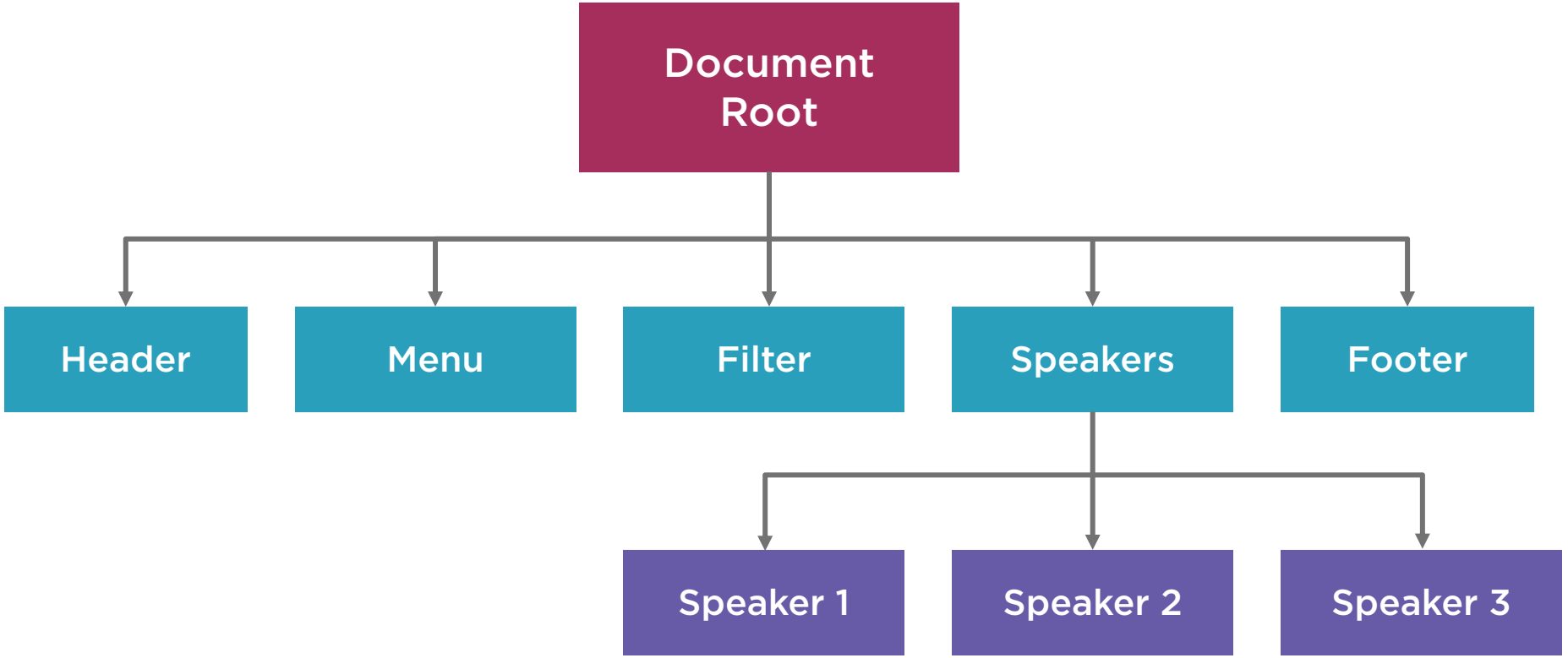


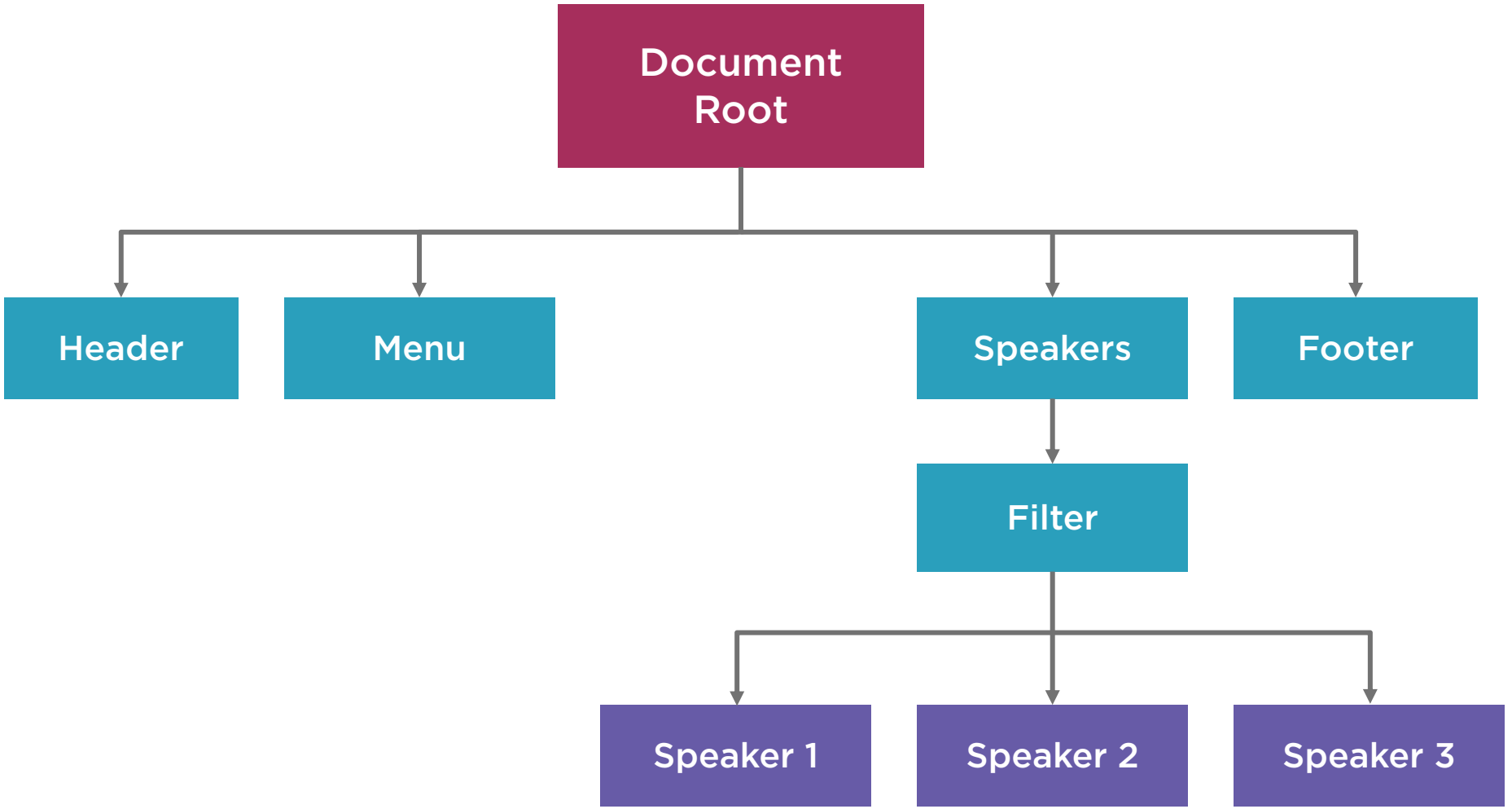

Document Root
Header
Menu
Filter
Speakers
Footer
Speaker 1
Speaker 2
Speaker 3

Deeper Dive into the Speaker Component






Document Root
Header
Menu
Filter
Speakers
Footer
Speaker 1
Speaker 2
Speaker 3


Document Root
Header
Menu
Speakers
Footer
Filter
Speaker 1
Speaker 2
Speaker 3

Common Layout Page

Different pages on site

Home

Speakers

Sessions

Schedules

Common components

Header

Menu

Footer

```
state: {
  speakers: [{speaker1},{speaker2},…]
}

setState({
  speakers: [{speaker1},{speaker2withNewValue},…]
});
```

REST

Representational state transfer (REST) is a software architectural style that defines a set of constraints to be used for creating Web services

HTTP VERB	URL Endpoint
GET	http://localhost:4000/speakers
PUT	http://localhost:4000/speakers/$ID

Async/await Coding Pattern

```
const response = await axios.get("http://..");
```

```
setSpeakers(response.data);
```

Async/await Coding Pattern

```
const response = await axios.get("http://..");
```

```
setSpeakers(response.data);
```

Page Loading Status

Loading

Success

Error

Async/await Coding Pattern

Code

```
const response = await axios.get());

setSpeakers(response.data);
```

Request Status

LOADING

SUCCESS

ERROR

Consolidating State Advantages

Multiple state variables change at once

Reuse state change logic

Consolidate state change logic

Reducer Defined

```
(previousState, action) => newState
```

Takeaways and Lookaheads

- Migrating from strawman code app
- Transition to real world app
- Reducers bring benefits
- Externalize reducers for reuse

- Coming up, isolating data access
  - Using HOCs
  - Using Render Props
  - Using the Context API