**React: application styling and state management**

State management: Redux, Mobx, Redux + Mobx

Styling react application:

Styling by development team choices: Bootstrap & Material UI

Styling provided by 3rd party: Styled Components

**Redux:**

<https://github.com/reactjs/redux/>

The whole state of your app is stored in an object tree inside a single *store*.

The only way to change the state tree is to emit an *action*, an object describing what happened.

To specify how the actions transform the state tree, you write pure *reducers*.

reactstyled

Branches:

redux

reduxvalues

**MobX:**

<https://github.com/mobxjs/mobx>

Branches:

mobx

mobxstore

Anything that can be derived from the application state, should be derived. Automatically.

which includes the UI, data serialization, server communication, etc.

**Mobx State Tree:**

<https://github.com/mobxjs/mobx-state-tree>

Central in MST (mobx-state-tree) is the concept of a *living tree*. The tree consists of mutable, but strictly protected objects enriched with *runtime type information*. In other words; each tree has a *shape* (type information) and *state* (data). From this living tree, immutable, structurally shared, snapshots are generated automatically.

Branches:

mobxstatetree

mobxstatetreetools

**Styling**

<http://andrewhfarmer.com/how-to-style-react/>

**React Bootstrap**

<https://react-bootstrap.github.io/components.html>

Branches:

reactbootstrap

**React Material UI**

<https://github.com/callemall/material-ui>

Branches:

reactmaterial

[http://www.codecovers.eu/materialadmin/ui](http://www.codecovers.eu/materialadmin/ui/buttons)

**React Styled Components**

<https://github.com/styled-components/styled-components>

<https://www.styled-components.com/docs/advanced#existing-css>

<https://www.styled-components.com/docs>

Branches:

reactstyled