Who hasn’t used Deliveroo?

https://roo.it/edds
Using React at Deliveroo

1. Why React?
2. Introducing it at Deliveroo
3. Growing with it
React is a component based templating library in JavaScript
Components have props passed into them, and state to save their... state

A component can’t modify its props only its state
Components are rendered when new props are passed in or state is changed.
class PageHeader extends React.Component {
    render () {
        return (<header>
            <h1>{this.props.pageTitle}</h1>
        </header>);
    }
}
Why did we choose to use React with our Rails stack at Deliveroo?
1. Server rendered views with progressive enhancement
Switching to a full server render using React has seen significant speed gains on every page we have converted.
1. Server rendered views with progressive enhancement
2. Separation of concerns
Ruby does
Getting objects
Presenting objects

React does
Rendering templates
1. Server rendered views with progressive enhancement
2. Separation of concerns
3. Behaviour described in templates
class PageHeader extends React.Component {
    bounce (event) {
        ...
    }

    render () {
        return (<header>
            <h1 onClick={this.bounce}>
                {this.props.pageTitle}
            </h1>
        </header>);
    }
}
1. Server rendered views with progressive enhancement
2. Separation of concerns
3. Behaviour described in templates
4. Testable server rendered views
The React contract states that given a set of props a component will always produce the same markup.
Using React at Deliveroo

1. Why React?
2. Introducing it at Deliveroo
3. Growing with it
Introduce the smallest change possible to bring as many people with you as possible
Every technology / dependency you introduction adds cost for the whole team, not just the initial cost
Prove the technology first going page by page, scale up when required
Vanilla React

No state stores, no extensions
Provide props for whole page render from the controller
# Gemfile

gem 'react-rails'

# Controller

def render component 'PageHeader',
  props: @props,
  layout: 'react-application'
Sticking with tools our engineers know:
  asset pipeline
  erb layouts
  RSpec
Asset Pipeline

./app/assets/javascripts
  ./react
    ./application.js
    ./page-header.js.jsx
RSpec

describe 'PageHeader', type: :component do
  context 'a basic header', :load_props do
    it { should have_text 'Hello LRUG!' }
  end
end
RSpec

# ./page-header fixtures.yaml
- title: a basic header
  props:
    title: Hello LRUG!
RSpec is also then used to test presenters and controller output.
Using React at Deliveroo

1. Why React?
2. Introducing it at Deliveroo
3. Growing with it
Sending all your templates to users on every page is expensive
Without a module system it’s hard to maintain component dependencies
We switched to `react-on-rails` gem to get Webpack for dependency management.
Had to leave RSpec for mocha but fixtures came with us because they were technology agnostic
Moving to react-on-rails initially gave us much worse performance
Adding therubyrunner and a node build pack to heroku made it fast
All engineers now need to maintain their NPM version
We’re still yet to start using a state store which is hurting us
Potential future enhancements:

Split front end to be its own app
Enhance client side to handle page navigation
Living style guide from fixture files
We’re Hiring!
Thank You! Questions?

Edd Sowden
Engineering Lead - Consumer Web
Deliveroo
@edds

https://roo.it/edds