Building the Right Product and Building It Right

A glimpse into Extreme Programming and Atomic Design

Rita Castro | SDC:LX | Remote React Summit 2021

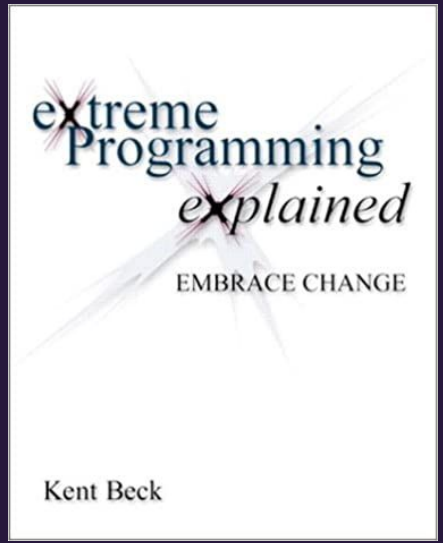
Hello ~~World~~ Remote React Summit!

Building the Right Product and Building It Right

SOFTWARE DEVELOPMENT CENTER : LISBON

Extreme Programming

XP is an agile software development framework that aims to produce higher quality software, and increase the quality of life for the development team



Values

Communication

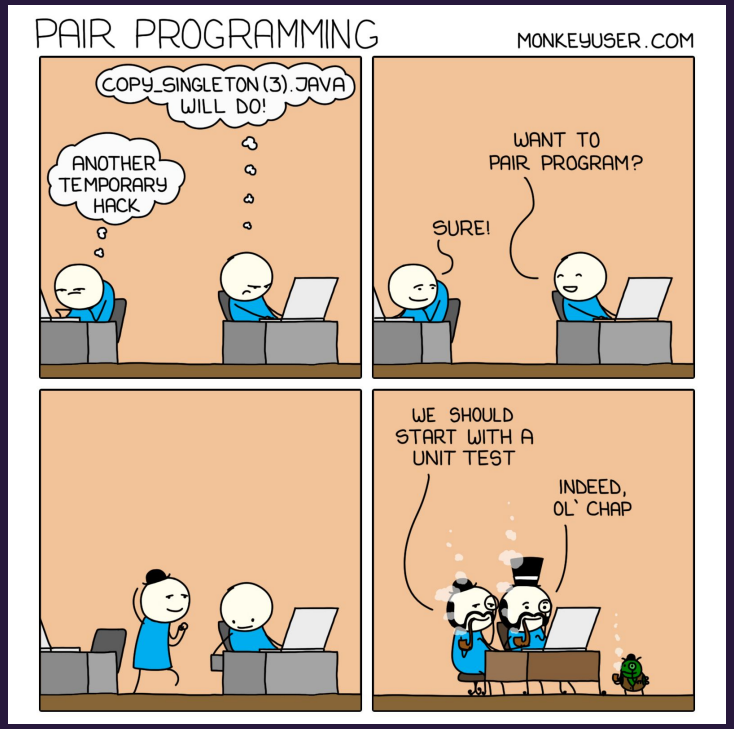
Simplicity

Feedback

Courage

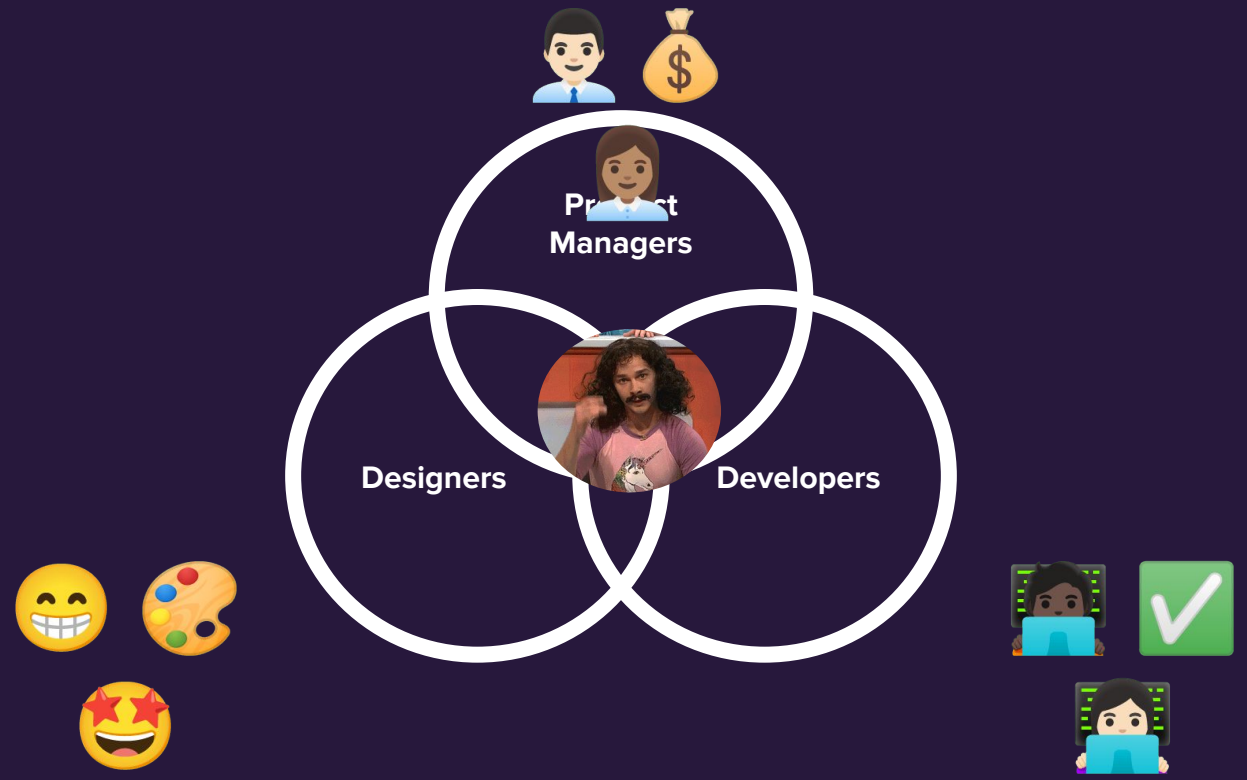
Respect

Practices

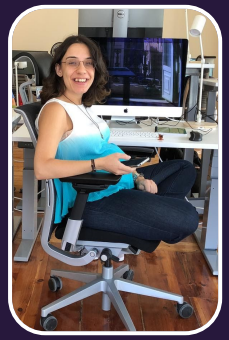

PAIR PROGRAMMING
MONKEYUSER.COM
COPY_SINGLETON(3).JAVA WILL DO!
ANOTHER TEMPORARY HACK
WANT TO PAIR PROGRAM?
SURE!
WE SHOULD START WITH A UNIT TEST
INDEED, OL` CHAP

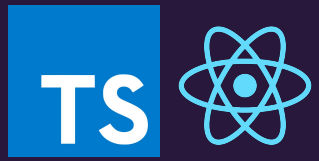

TEST
TEST
DRIVEN
DRIVEN
DEVELOPMENT
DEVELOPMENT
TEST DRIVEN DEVELOPMENT
JUST HAVE FAITH
imgflip.com

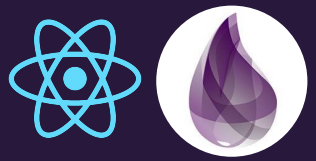
Balanced Teams



Me in the SDC:LX



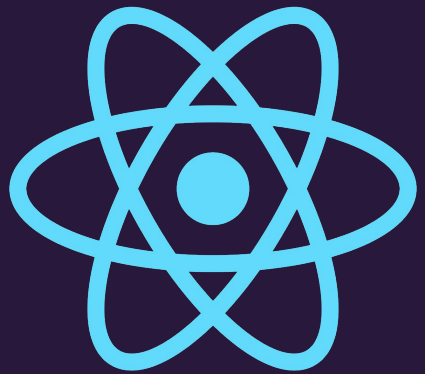




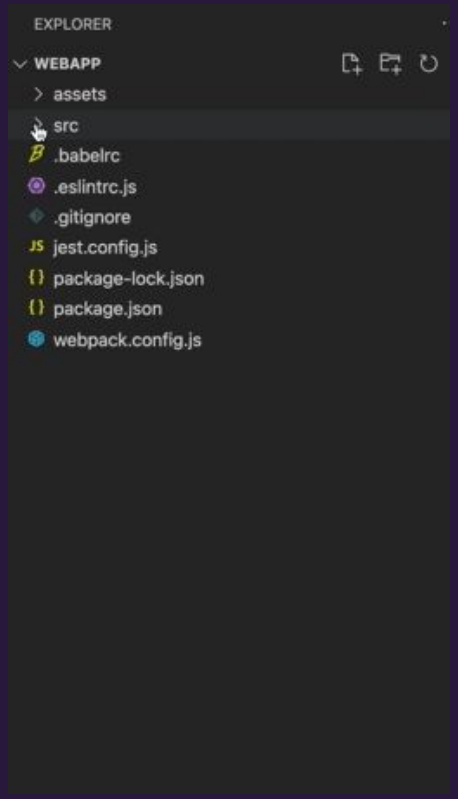
Why React?

“making it painless to create interactive UIs”

“encapsulated components … then compose them to make complex UIs”

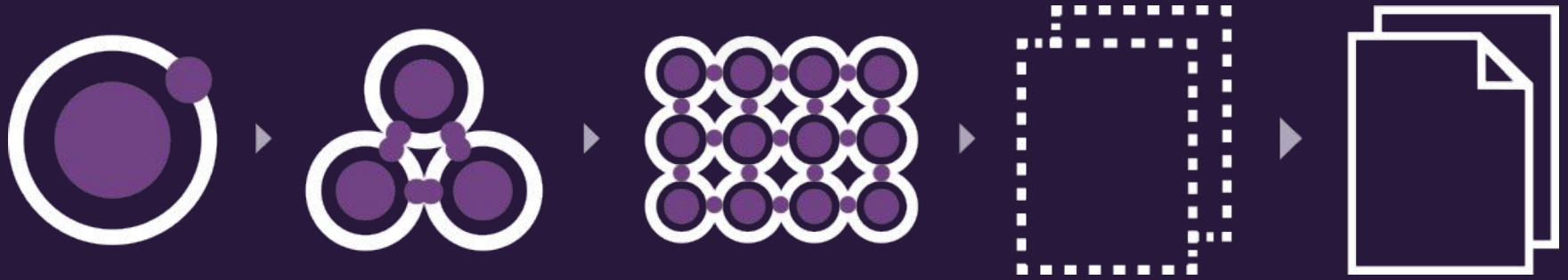


Feature driven development



Atomic Design

It “details all that goes into creating and maintaining robust design systems, allowing you to roll out higher quality, more consistent UIs faster than ever before".

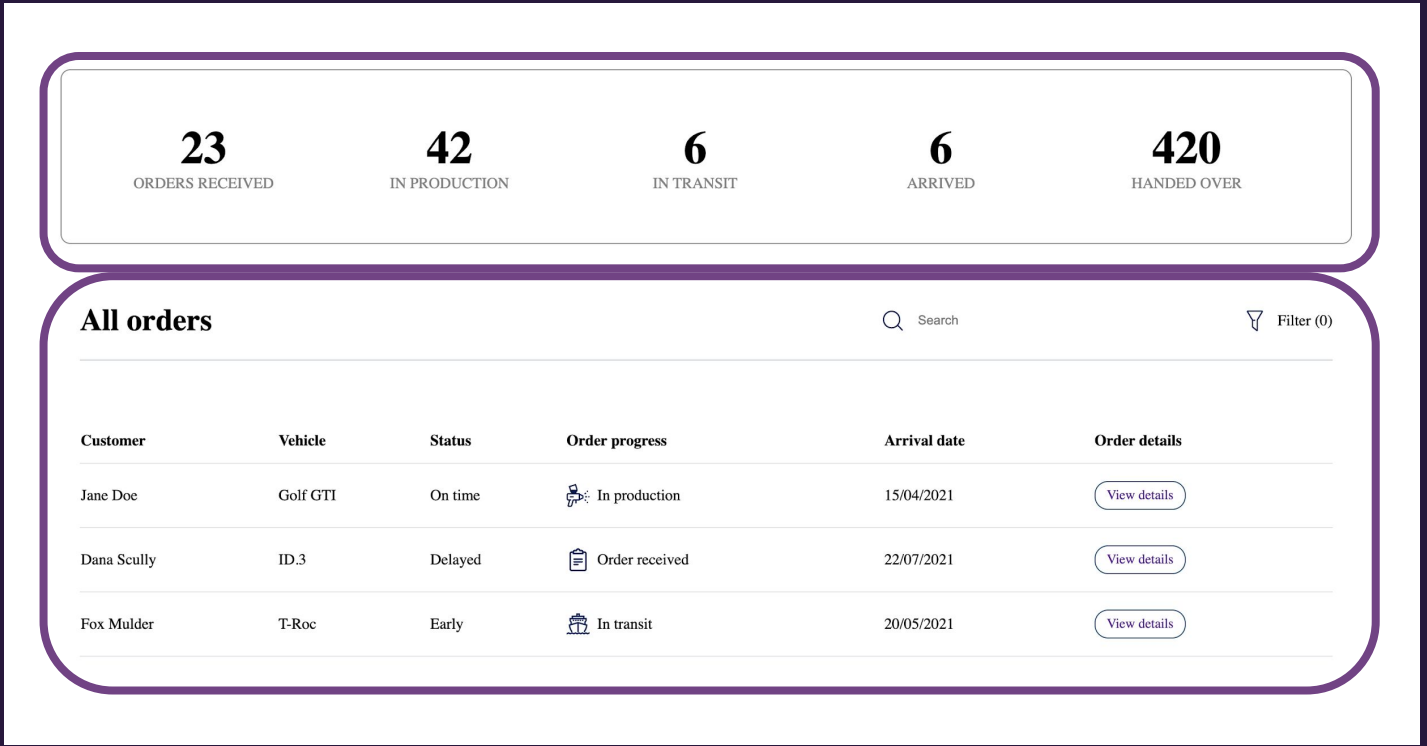


[Brad Frost, https://bradfrost.com/blog/post/atomic-web-design/]

And now, this...

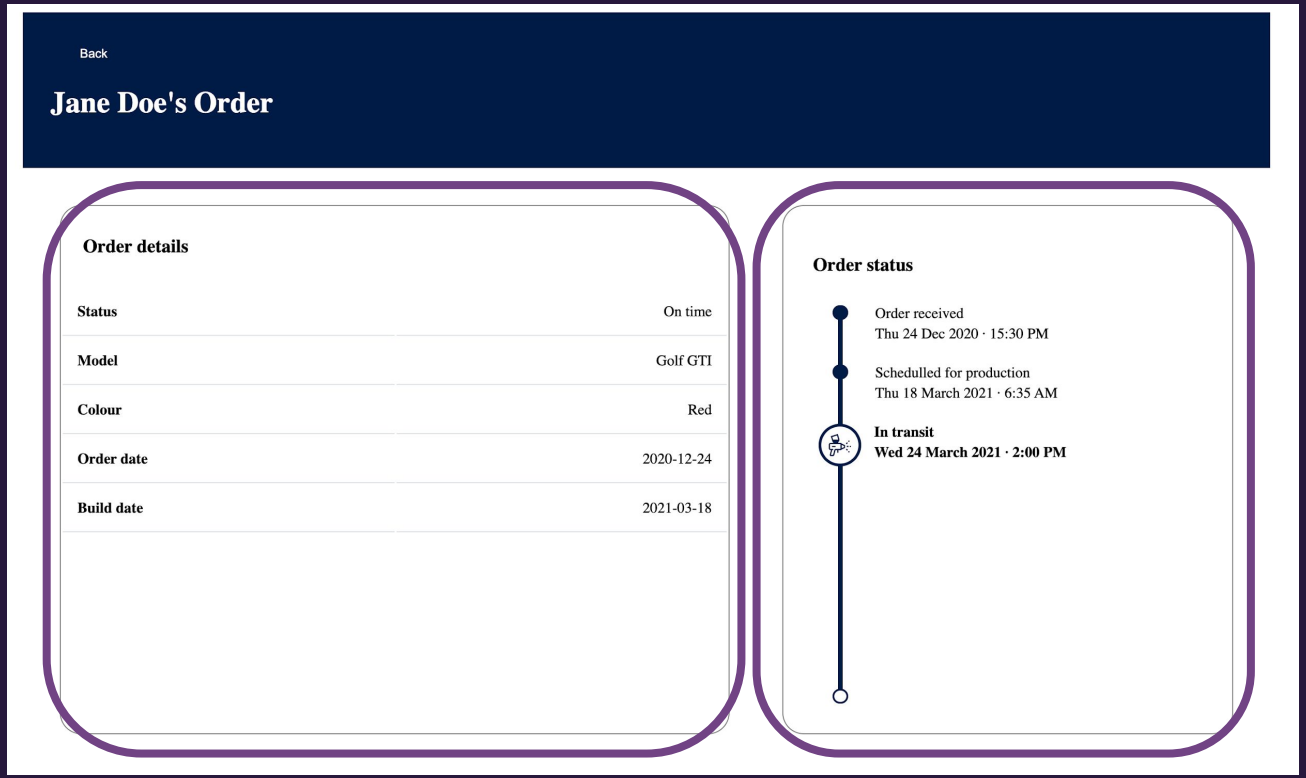
User Interviews with Dealers

Dashboard



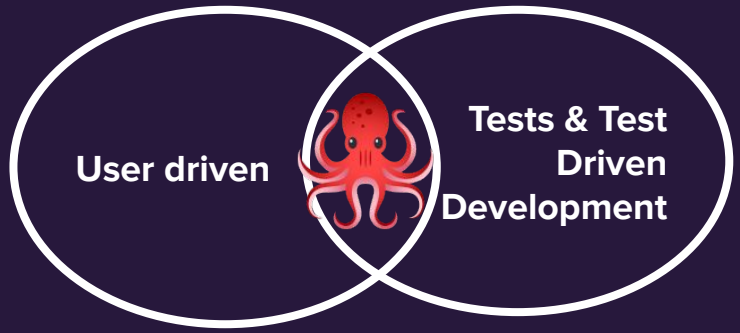
Mocked designs

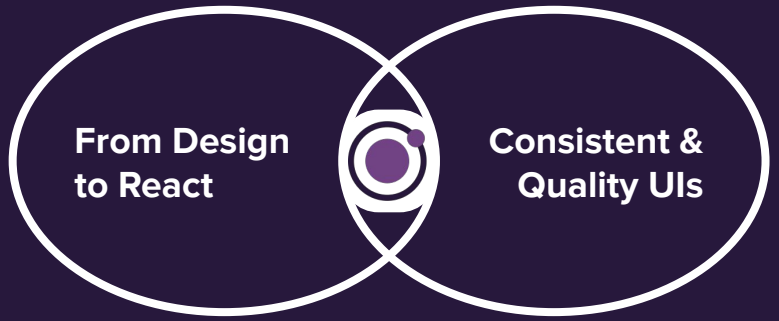
Details Page



Mocked designs

Remember this?





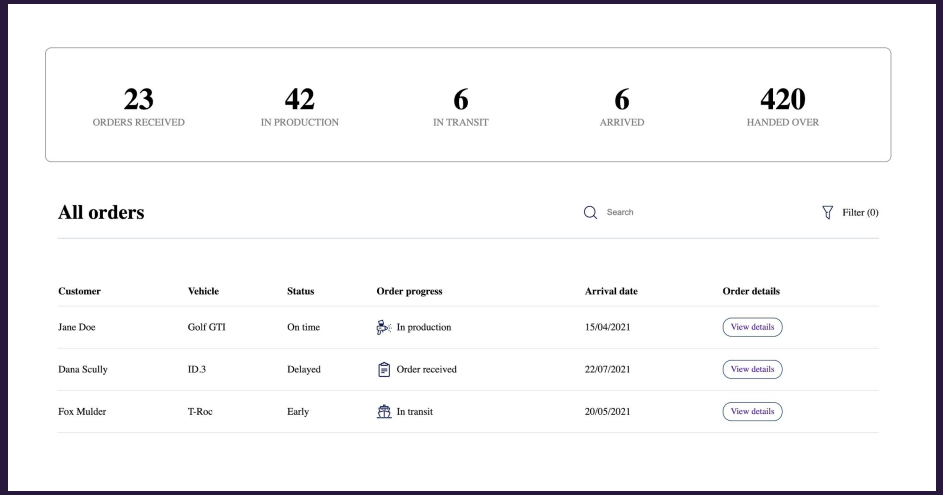
It becomes this...

Write a failing test

… but which one?

Start small...

From Designs to Atomic components



- Unbreakable components
- Base styles from the UI

View details

15/04/2021

ID.3 Style

`ViewDetails` as an Atom

```
import React from "react"
import { render, screen } from "@testing-library/react"
import ViewDetailsButton from "./view-details-button"
import { Link } from "react-router-dom"

jest.mock("react-router-dom", () => ({
    ...jest.requireActual("react-router-dom"),
    Link: jest.fn().mockImplementation(({ children }) => children)
}))

describe("ViewDetails button", () => {
    it("shows the 'See Details' that will takes us somewhere", () => {
        render(
            <ViewDetailsButton to="/take-me-to-react-summit-in-amsterdam" />
        )
        expect(screen.getByText("View details")).toBeInTheDocument()

        expect(Link).toHaveBeenCalledWith(expect.objectContaining(
            { to: "/take-me-to-react-summit-in-amsterdam" }
        ), {})
    })
})
```

```
1  import React from "react"
2
3  const ViewDetailsButton = () => {
4
5      return null
6  }
7
8  export default ViewDetailsButton
9
```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | 1: node

```
FAIL src/ui/atoms/view-details-button/view-details-button-with-mocks.test.jsx
  ViewDetails button
    × shows the 'See Details' that will takes us somewhere (19 ms)

  ● ViewDetails button › shows the 'See Details' that will takes us somewhere

    TestingLibraryElementError: Unable to find an element with the text: View details. This could be because the text is broken up by multiple elements. In this case, you can provide a function for your text matcher to make your matcher more flexible.

    <body>
      <div />
    </body>

      15 |             <ViewDetailsButton to="/take-me-to-react-summit-in-amsterdam" />
      16 |         )
    > 17 |         expect(screen.getByText("View details")).toBeInTheDocument()
         |                ^
      18 |
      19 |         expect(Link).toHaveBeenCalledWith(expect.objectContaining(
      20 |             { to: "/take-me-to-react-summit-in-amsterdam" }

      at Object.getElementError (node_modules/@testing-library/dom/dist/config.js:37:19)
      at node_modules/@testing-library/dom/dist/query-helpers.js:90:38
      at node_modules/@testing-library/dom/dist/query-helpers.js:62:17
      at getByText (node_modules/@testing-library/dom/dist/query-helpers.js:111:19)
      at Object.<anonymous> (src/ui/atoms/view-details-button/view-details-button-with-mocks.test.jsx:17:23)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
Snapshots:   0 total
Time:        2.767 s, estimated 3 s
Ran all test suites matching /src\/ui\/atoms\/view-details-button\/view-details-button-with-mocks\.test\.jsx/i.
```

```
1  import React from "react"
2
3  const ViewDetailsButton = () => {
4
5      return <p>View details</p>      You, seconds ago • Uncommitted c
6  }
7
8  export default ViewDetailsButton
9
```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | 1: node

```
FAIL src/ui/atoms/view-details-button/view-details-button-with-mocks.test.jsx
  ViewDetails button
    × shows the 'See Details' that will takes us somewhere (23 ms)

  ● ViewDetails button › shows the 'See Details' that will takes us somewhere

    expect(jest.fn()).toHaveBeenCalledWith(...expected)

    Expected: ObjectContaining {"to": "/take-me-to-react-summit-in-amsterdam"}, {}

    Number of calls: 0

      17 |         expect(screen.getByText("View details")).toBeInTheDocument()
      18 |
    > 19 |       expect(Link).toHaveBeenCalledWith(expect.objectContaining(
         |                    ^
      20 |           { to: "/take-me-to-react-summit-in-amsterdam" }
      21 |       ), {})
      22 |     })

      at Object.<anonymous> (src/ui/atoms/view-details-button/view-details-button-with-mocks.test.jsx:19:22)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
Snapshots:   0 total
Time:        4.567 s
Ran all test suites matching /src\/ui\/atoms\/view-details-button\/view-details-button-with-mocks\.test\.jsx/i.

Watch Usage: Press w to show more.
```

```jsx
import React from "react"
import { Link } from "react-router-dom"

const ViewDetailsButton = ({ to }) => {

    return <Link className="sassy-link" to={to}>
        View details
    </Link>
}

export default ViewDetailsButton
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL 1: node

```
PASS src/ui/atoms/view-details-button/view-details-button-with-mocks.test.jsx
  ViewDetails button
    √ shows the 'See Details' that will takes us somewhere (21 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        3.291 s
Ran all test suites matching /src\/ui\/atoms\/view-details-button\/view-detail

Watch Usage: Press w to show more.
```

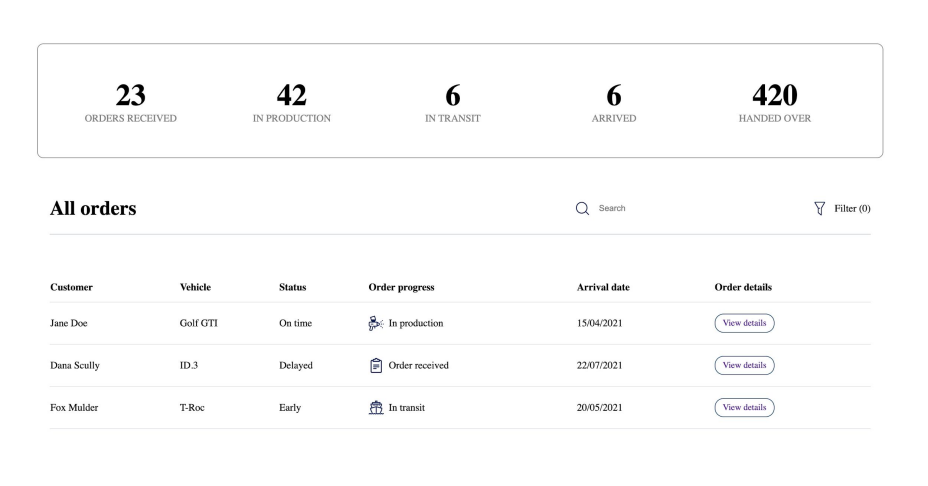
`ViewDetails` as an Atom - revisited

```
import React from "react"
import { render, screen } from "@testing-library/react"
import ViewDetailsButton from "./view-details-button"
import { MemoryRouter } from "react-router-dom"

describe("ViewDetails button", () => {
    it("shows the 'See Details' that will takes us somewhere", () => {

        render(
            <MemoryRouter initialEntries={["/lisbon"]}>
                <ViewDetailsButton to="/take-me-to-react-summit-in-amsterdam" />
            </MemoryRouter>
        )

        const viewDetails = screen.getByRole("link", { name: "View details" })
        expect(viewDetails).toBeInTheDocument()
        expect(viewDetails).toHaveAttribute(
            "href", "/take-me-to-react-summit-in-amsterdam")

        expect(viewDetails).toHaveClass("sassy-link")
    })
})
```

You do you

- ➔ There is no right or wrong
- ➔ Team’s decision

From Designs to Atomic components



23	42	6	6	420
ORDERS RECEIVED	IN PRODUCTION	IN TRANSIT	ARRIVED	HANDED OVER

All orders Search Filter (0)

Customer	Vehicle	Status	Order progress	Arrival date	Order details
Jane Doe	Golf GTI	On time	In production	15/04/2021	View details
Dana Scully	ID.3	Delayed	Order received	22/07/2021	View details
Fox Mulder	T-Roc	Early	In transit	20/05/2021	View details

- Atoms bonded together
- Bring functionality to the UI

Search

In transit

The `Progress` molecule...

src > ui > molecules > progress > progress.test.jsx > ...

```jsx
import React from "react"
import { render, screen } from "@testing-library/react"
import Progress from "./progress"
import InTransitIcon from "../../atoms/icons/in-transit"

jest.mock("../../atoms/icons/in-transit", () => {
    return {
        __esModule: true,
        default: jest.fn().mockReturnValue(null)
    }
})

describe("Progress molecule", () => {
    it("renders the visual elements when status is 'In transit'", () => {
        render(
            <Progress />
        )

        expect(screen.getByText("In transit")).toBeInTheDocument()

        expect(InTransitIcon).toHaveBeenCalled()
    })

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL 1: node

```
FAIL src/ui/molecules/progress/progress.test.jsx
  Progress molecule
    × renders the visual elements when status is 'In transit' (6 ms)

  ● Progress molecule › renders the visual elements when status is 'In transit'

    TestingLibraryElementError: Unable to find an element with the text: In transit. This could be because the t
ext is broken up by multiple elements. In this case, you can provide a function for your text matcher to make yo
ur matcher more flexible.

    <body>
      <div>
        <div />
      </div>
    </body>
```

..is composed of two Atoms

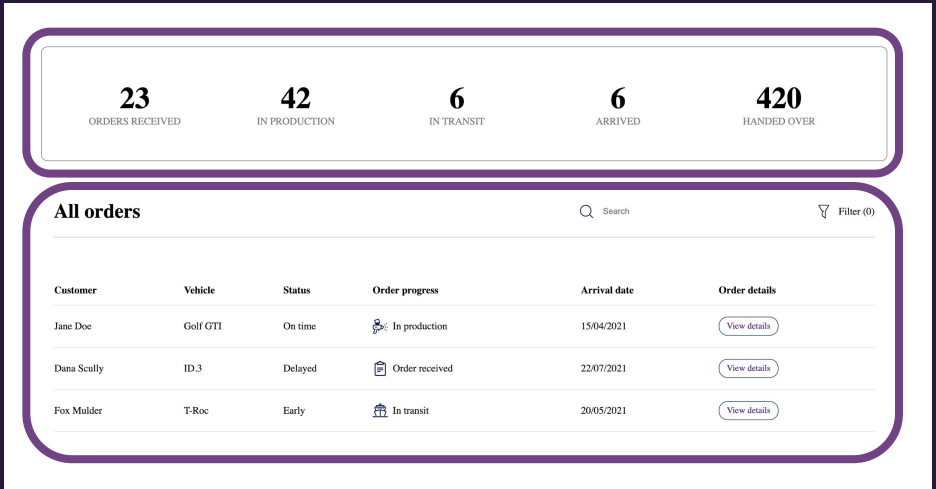
src > ui > molecules > progress > progress.jsx > ...

```jsx
import React from "react"
import InTransitIcon from "../../atoms/icons/in-transit"
import Text from "../../atoms/text/text"

const Progress = () => {

    return (
        <div>
            <InTransitIcon />
            <Text>In transit</Text>
        </div>
    )
}

export default Progress
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL 1: node

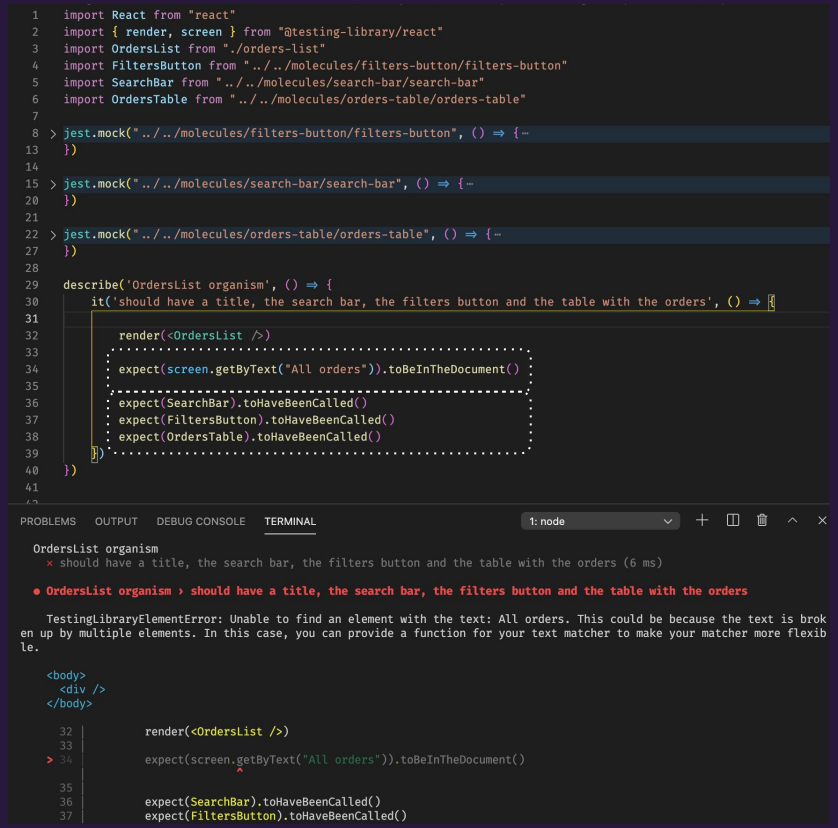
```
PASS src/ui/molecules/progress/progress.test.jsx
  Progress molecule
    √ renders the visual elements when status is 'In transit' (24 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        1.402 s
Ran all test suites matching /src\/ui\/molecules\/progress\/progress\.test\.jsx/i.

Watch Usage: Press w to show more.
```

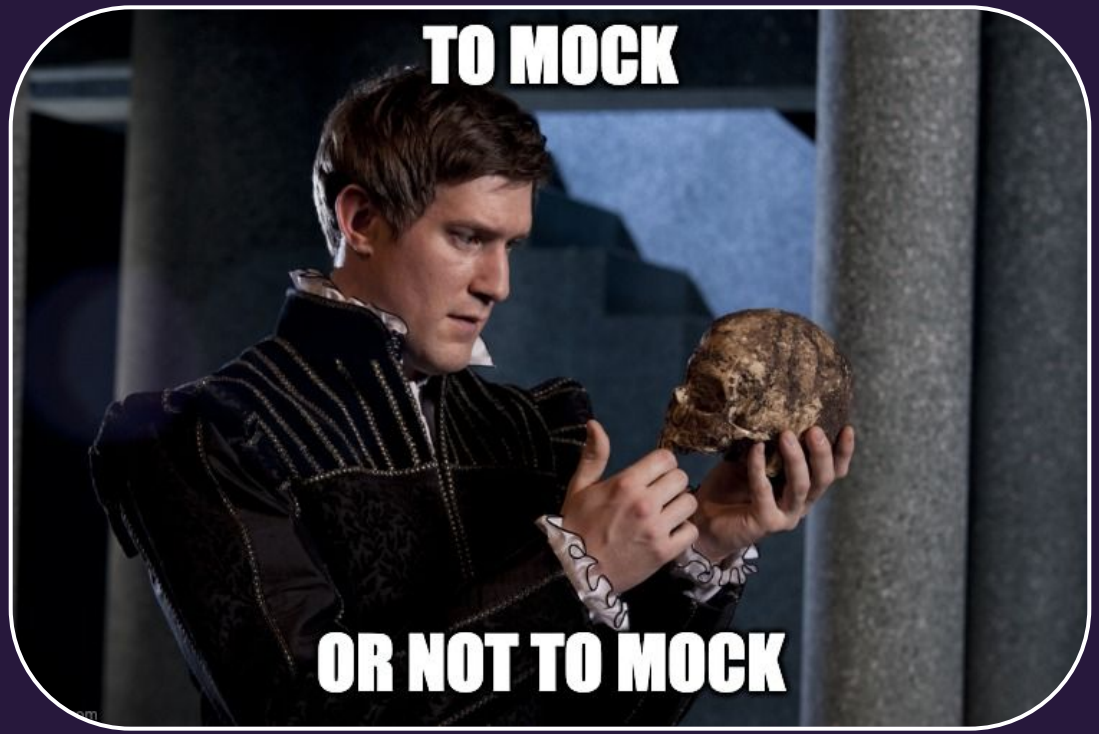
From Designs to Atomic components



- A flock of atoms, molecules and even organisms!
- ***User Value***

`OrderTable` is an Organism!




TO MOCK
OR NOT TO MOCK

Pros & Cons on Mocking

👍 Encapsulation
No need to know details

👍 Clear view of elements used

👍 No cheating!!!
Bazinga vs <Text>Bazinga</Text>

👍 Easy to onboard new team members

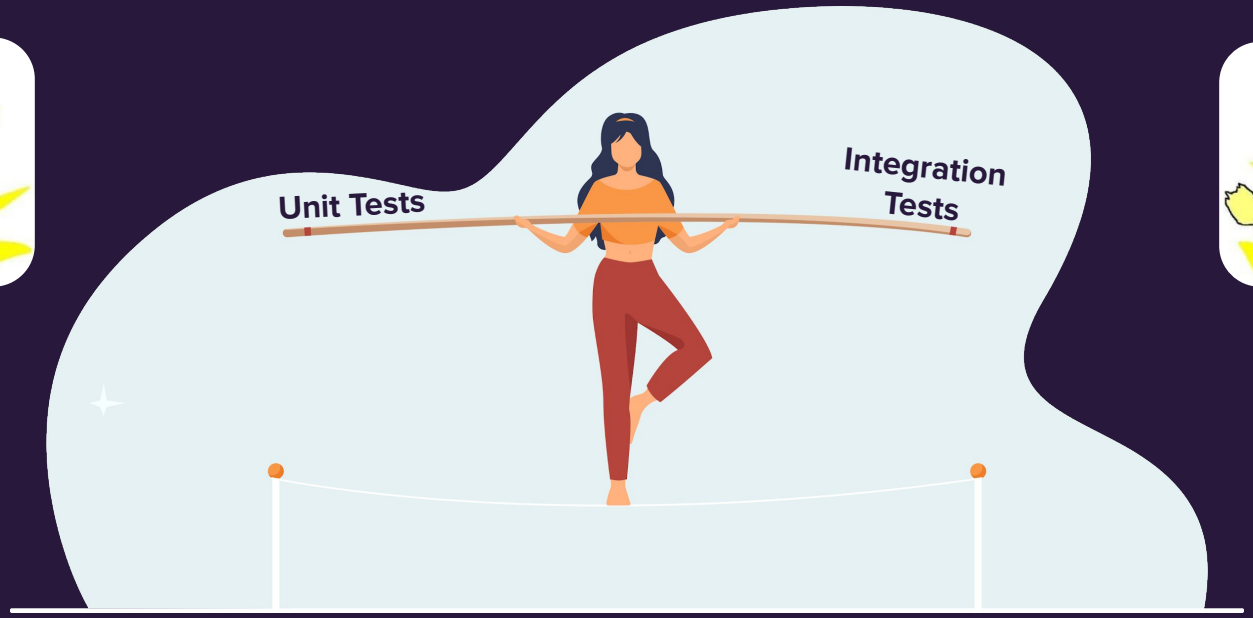
👎 Further away from User experience

👎 ““Shallow”” render going on

👎 Might duplicate tests

👎 Mocks can be tricky
Specially with third party libs

Find your balance

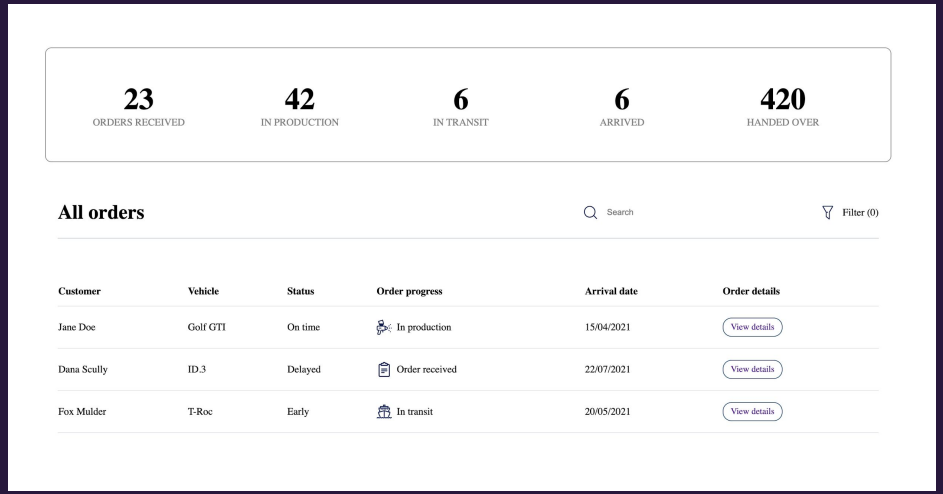


The more your tests resemble the way your software is used, the more confidence they can give you.

[Kent C. Dodds, hhttps://twitter.com/kentcdodds/status/977018512689455106?lang=en

```
import React from "react"
import { MemoryRouter, Route, Switch } from "react-router"
import { render, screen } from "@testing-library/react"
import userEvent from '@testing-library/user-event'
import OrdersList from "./orders-list"
import { getOrders } from "../../../services/get-orders"

jest.mock("../../../services/get-orders", () => ({
    getOrders: jest.fn()
}))

describe("OrdersList organism", () => {
    it("shows the order list and we can click to 'View Details'", () => {

        getOrders.mockReturnValue([
            {
                id: 1024,
                name: "Rita Castro",
                vehicle: "T-Roc",
                status: "delayed",
                progress: "order received",
                arrivalDate: "2021-07-24",
            }
        ])

        render(
            <MemoryRouter initialEntries={["/"]}>
                <Switch>
                    <Route exact path="/" render={() => <OrdersList />} />
                    <Route path="/orders/:orderId" render={() => <div>Stub Details Page</div>} />
                </Switch>
            </MemoryRouter>
        )

        expect(screen.getByRole("heading", { name: "All orders" })).toBeInTheDocument()

        expect(screen.getByText("Rita Castro")).toBeInTheDocument()
        // same goes for the other elements in the table

        const viewDetailsButton = screen.getByRole("link", { name: "View details" })
        userEvent.click(viewDetailsButton)

        expect(screen.getByText("Stub Details Page")).toBeInTheDocument()
    })
})
```

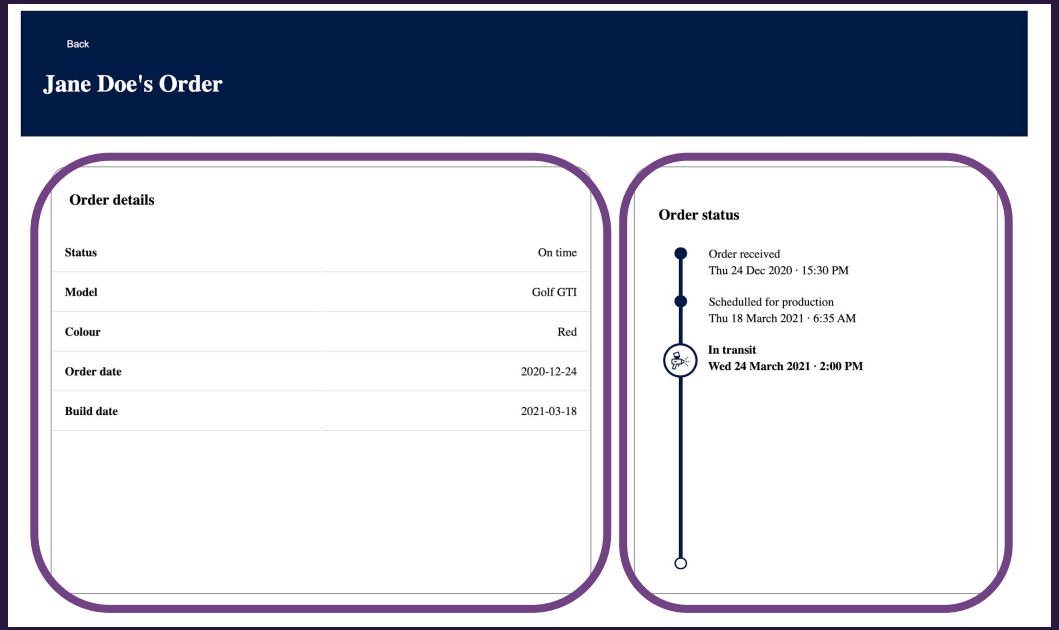
From Designs to Atomic components

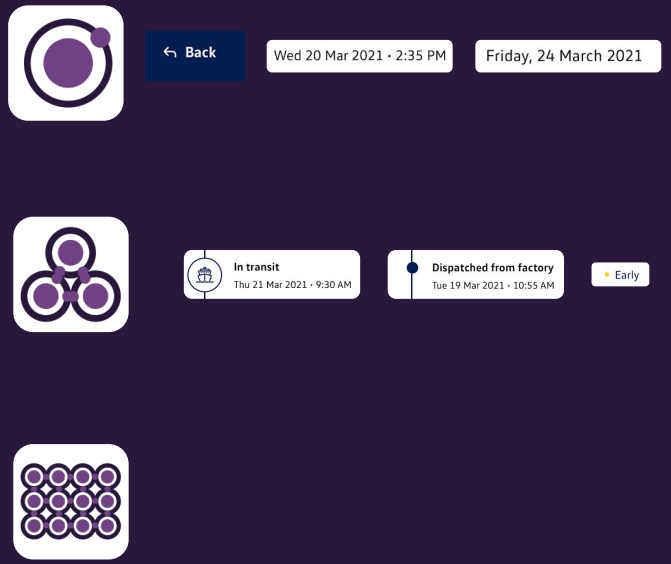


&

- Define a layout
- Place organisms there

From [more] Designs to [more] Atomic components





The Dealer’s journey

```tsx
import React from "react"
import { render, screen, waitFor, within } from "@testing-library/react"
import userEvent from "@testing-library/user-event"
import VehicleOrderTracker from "./vehicle-order-tracker"

describe("Vehicle order tracker for Remote React Summit 2021", () => {

    it("renders the Dashboard and we can 'View details' of an Order and come 'Back' to the Dashboard", async () => {

        render(<VehicleOrderTracker />)

        const dashboard = screen.getByRole("region", { name: "Dashboard" })
        expect(dashboard).toBeInTheDocument()

        const overviewTabs = within(dashboard).getByRole("tablist")
        expect(overviewTabs).toBeInTheDocument()

        expect(within(dashboard).getByText("All orders")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Search for an order")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Filter orders")).toBeInTheDocument()

        const scullysOrder = within(dashboard).getByLabelText("Dana Scully's order")
        expect(scullysOrder).toBeInTheDocument()

        const viewDetailsButton = within(scullysOrder).getByRole("link", { name: "View details" })
        userEvent.click(viewDetailsButton)

        expect(screen.queryByRole("region", { name: "Dashboard" })).not.toBeInTheDocument()
        expect(screen.getByRole("region", { name: "Order details" })).toBeInTheDocument()

        const backButton = screen.getByRole("button", { name: "Back" })
        expect(backButton).toBeInTheDocument()

        userEvent.click(backButton)

        await waitFor(() => {
            expect(screen.getByText("All orders")).toBeInTheDocument()
        })
    })
})
```

```tsx
import React from "react"
import OrdersList from "../../organisms/orders-list/orders-list"
import OverviewTabs from "../../organisms/overview-tabs/overview-tabs"

const Dashboard = () => {

    return (
        <div role="region" aria-label="Dashboard">
            <OverviewTabs />

            <OrdersList />
        </div>
    )
}

export default Dashboard
```

The Dealer’s journey

```
import React from "react"
import { render, screen, waitFor, within } from "@testing-library/react"
import userEvent from "@testing-library/user-event"
import VehicleOrderTracker from "./vehicle-order-tracker"

describe("Vehicle order tracker for Remote React Summit 2021", () => {

    it("renders the Dashboard and we can 'View details' of an Order and come 'Back' to the Dashboard", async () => {

        render(<VehicleOrderTracker />)

        const dashboard = screen.getByRole("region", { name: "Dashboard" })
        expect(dashboard).toBeInTheDocument()

        const overviewTabs = within(dashboard).getByRole("tablist")
        expect(overviewTabs).toBeInTheDocument()

        expect(within(dashboard).getByText("All orders")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Search for an order")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Filter orders")).toBeInTheDocument()

        const scullysOrder = within(dashboard).getByLabelText("Dana Scully's order")
        expect(scullysOrder).toBeInTheDocument()

        const viewDetailsButton = within(scullysOrder).getByRole("link", { name: "View details" })
        userEvent.click(viewDetailsButton)

        expect(screen.queryByRole("region", { name: "Dashboard" })).not.toBeInTheDocument()
        expect(screen.getByRole("region", { name: "Order details" })).toBeInTheDocument()

        const backButton = screen.getByRole("button", { name: "Back" })
        expect(backButton).toBeInTheDocument()

        userEvent.click(backButton)

        await waitFor(() => {
            expect(screen.getByText("All orders")).toBeInTheDocument()
        })
    })
})
```

```
import React, { useEffect, useState } from "react"
import { getOrders } from "../../../services/get-orders"
import FormattedDateNumbers from "../../../atoms/formatted-date/numbers/formatted-date-numbers"
import ViewDetailsButton from "../../../atoms/view-details-button/view-details-button"

const OrdersTable = () => {

    const [orders, setOrders] = useState([])

    useEffect(() => setOrders(getOrders), [])

    return (
        <table aria-label="Orders list">
            <thead>
                <tr>
                    <th>Customer</th>
                    <th>Vehicle</th>
                    <th>Status</th>
                    <th>Order progress</th>
                    <th>Arrival date</th>
                    <th>Order details</th>
                </tr>
            </thead>
            <tbody>
                {
                    orders.map(order =>
                        <tr key={order.id} aria-label={`${order.name}'s order`}>
                            <td>{order.name}</td>
                            <td>{order.vehicle}</td>
                            <td>{order.status}</td>
                            <td>{order.progress}</td>
                            <td><FormattedDateNumbers value={order.arrivalDate} /></td>
                            <td><ViewDetailsButton to={`/orders/${order.id}`} /></td>
                        </tr>
                    )
                }
            </tbody>
        </table>
    )
}
```

The Dealer’s journey

```
import React from "react"
import { render, screen, waitFor, within } from "@testing-library/react"
import userEvent from "@testing-library/user-event"
import VehicleOrderTracker from "./vehicle-order-tracker"

describe("Vehicle order tracker for Remote React Summit 2021", () => {

    it("renders the Dashboard and we can 'View details' of an Order and come 'Back' to the Dashboard", async () => {

        render(<VehicleOrderTracker />)

        const dashboard = screen.getByRole("region", { name: "Dashboard" })
        expect(dashboard).toBeInTheDocument()

        const overviewTabs = within(dashboard).getByRole("tablist")
        expect(overviewTabs).toBeInTheDocument()

        expect(within(dashboard).getByText("All orders")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Search for an order")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Filter orders")).toBeInTheDocument()

        const scullysOrder = within(dashboard).getByLabelText("Dana Scully's order")
        expect(scullysOrder).toBeInTheDocument()

        const viewDetailsButton = within(scullysOrder).getByRole("link", { name: "View details" })
        userEvent.click(viewDetailsButton)

        expect(screen.queryByRole("region", { name: "Dashboard" })).not.toBeInTheDocument()
        expect(screen.getByRole("region", { name: "Order details" })).toBeInTheDocument()

        const backButton = screen.getByRole("button", { name: "Back" })
        expect(backButton).toBeInTheDocument()

        userEvent.click(backButton)

        await waitFor(() => {
            expect(screen.getByText("All orders")).toBeInTheDocument()
        })
    })
})
```

```
import React from "react"
import { BrowserRouter, Link, Redirect, Route, Switch } from "react-router-dom"
import Dashboard from "./ui/pages/dashboard/dashboard"
import OrderDetails from "./ui/pages/order-details/order-details"

const VehicleOrderTracker = () => {

    return <BrowserRouter>
        <header>
            <Link to="/">
                <img alt="Home" src="assets/images/mandala.png" height="42" width="42" />
            </Link>
        </header>

        <main>
            <Switch>
                <Route path="/dashboard" render={() => <Dashboard />} />
                <Route path={`/orders/:orderId`} render={() => <OrderDetails />} />
                <Redirect exact from={"/"} to={"/dashboard"} />
            </Switch>
        </main>

        <footer>
            <p>Rita Castro | 2021</p>
        </footer>
    </BrowserRouter>
}

export default VehicleOrderTracker
```

The Dealer’s journey

```
import React from "react"
import { render, screen, waitFor, within } from "@testing-library/react"
import userEvent from "@testing-library/user-event"
import VehicleOrderTracker from "./vehicle-order-tracker"

describe("Vehicle order tracker for Remote React Summit 2021", () => {

    it("renders the Dashboard and we can 'View details' of an Order and come 'Back' to the Dashboard", async () => {

        render(<VehicleOrderTracker />)

        const dashboard = screen.getByRole("region", { name: "Dashboard" })
        expect(dashboard).toBeInTheDocument()

        const overviewTabs = within(dashboard).getByRole("tablist")
        expect(overviewTabs).toBeInTheDocument()

        expect(within(dashboard).getByText("All orders")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Search for an order")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Filter orders")).toBeInTheDocument()

        const scullysOrder = within(dashboard).getByLabelText("Dana Scully's order")
        expect(scullysOrder).toBeInTheDocument()

        const viewDetailsButton = within(scullysOrder).getByRole("link", { name: "View details" })
        userEvent.click(viewDetailsButton)

        expect(screen.queryByRole("region", { name: "Dashboard" })).not.toBeInTheDocument()
        expect(screen.getByRole("region", { name: "Order details" })).toBeInTheDocument()

        const backButton = screen.getByRole("button", { name: "Back" })
        expect(backButton).toBeInTheDocument()

        userEvent.click(backButton)

        await waitFor(() => {
            expect(screen.getByText("All orders")).toBeInTheDocument()
        })
    })
})
```

```
import React, { useEffect, useState } from "react"
import { useParams } from "react-router"
import { getOrderDetails } from "../../../services/get-orders"
import BackButton from "../../atoms/back-button/back-button"

const OrderDetails = () => {
    const params = useParams()

    const [order, setOrder] = useState()

    useEffect(() => {
        setOrder(getOrderDetails(params.orderId))
    }, [])

    return (
        <div role="region" aria-label="Order details">
            <BackButton />
            <OrderSummary order={order} />
            <OrderStatusTracker order={order} />
        </div>
    )
}

export default OrderDetails
```

The Dealer’s journey

```
import React from "react"
import { render, screen, waitFor, within } from "@testing-library/react"
import userEvent from "@testing-library/user-event"
import VehicleOrderTracker from "./vehicle-order-tracker"

describe("Vehicle order tracker for Remote React Summit 2021", () => {

    it("renders the Dashboard and we can 'View details' of an Order and come 'Back' to the Dashboard", async () => {

        render(<VehicleOrderTracker />)

        const dashboard = screen.getByRole("region", { name: "Dashboard" })
        expect(dashboard).toBeInTheDocument()

        const overviewTabs = within(dashboard).getByRole("tablist")
        expect(overviewTabs).toBeInTheDocument()

        expect(within(dashboard).getByText("All orders")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Search for an order")).toBeInTheDocument()
        expect(within(dashboard).getByLabelText("Filter orders")).toBeInTheDocument()

        const scullysOrder = within(dashboard).getByLabelText("Dana Scully's order")
        expect(scullysOrder).toBeInTheDocument()

        const viewDetailsButton = within(scullysOrder).getByRole("link", { name: "View details" })
        userEvent.click(viewDetailsButton)

        expect(screen.queryByRole("region", { name: "Dashboard" })).not.toBeInTheDocument()
        expect(screen.getByRole("region", { name: "Order details" })).toBeInTheDocument()

        const backButton = screen.getByRole("button", { name: "Back" })
        expect(backButton).toBeInTheDocument()

        userEvent.click(backButton)

        await waitFor(() => {
            expect(screen.getByText("All orders")).toBeInTheDocument()
        })
    })
})
```

To deliver high quality software with consistent and solid UIs faster than ever before.

Any day of the week.
Including Fridays.

Thank you!

https://github.com/**ritamcastro**

ritamcastro@gmail.com

https://www.linkedin.com/in/**ritamcastro**