HOW **THE HELL** A REACT APP WORKS

A **simple** guide* about
React basics & state management

*Using TypeScript!

about me

@huan_mp

6+ years working on cutting-edge front technologies?

Nope. Definitely, **not** a Front End developer

But, curious and eager to learn new things **:)**

WHAT'S THE POINT?

Understand the basics
How, same things, can be done differently
Use this as a simple guide

OUTLINE

What's react?
Class components
Functional components
Component lifecycle and hooks
Advance use of state
  - Redux (+ workshop)
  - Nested stores
  - Redux Sagas
  - React query
Wrap up!

day 1

OUTLINE

What's react?
Class components
Functional components
Component lifecycle and hooks
Advance use of state
  - Redux (+ workshop)
  - Nested stores
  - Redux Sagas
  - React query
Wrap up!

day 2

OUTLINE

What's react?
Class components
Functional components
Component lifecycle and hooks
Advance use of state
  - Redux (+ workshop)
  - Nested stores
  - Redux Sagas
  - React query
Wrap up!

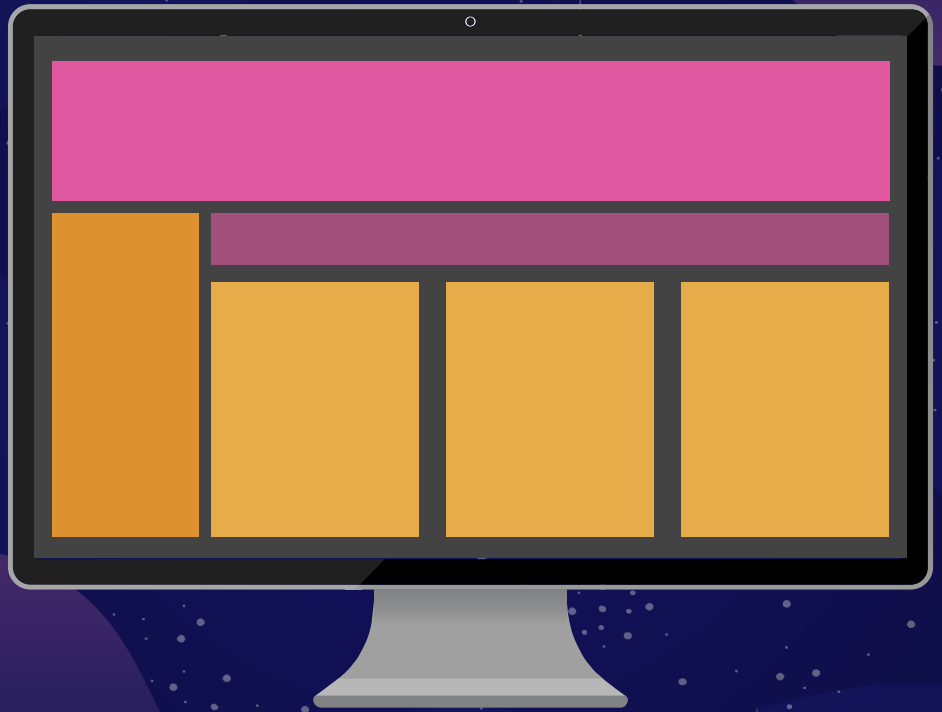
00

WHAT'S REACT?

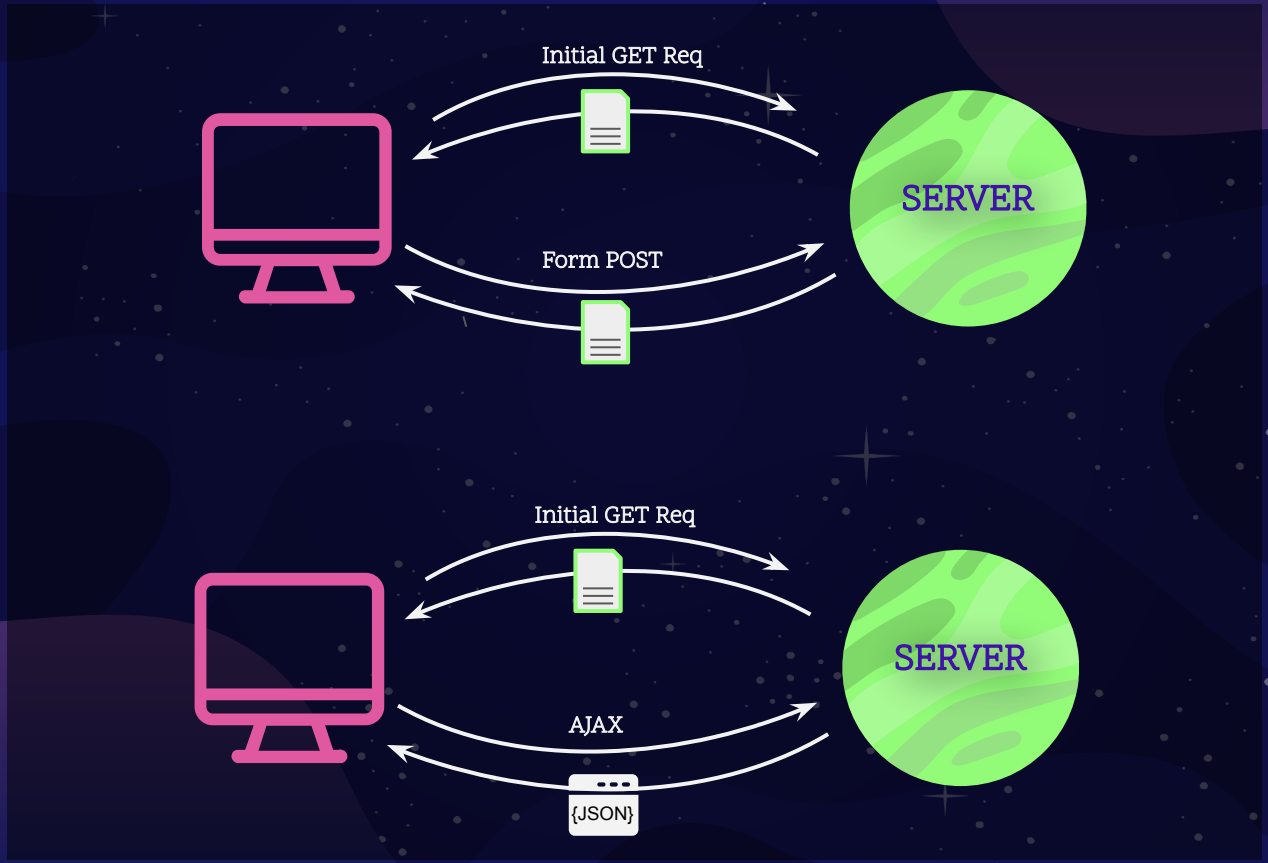
"A **JavaScript library** for building **user interfaces**"

-reactjs.org

AHAM! USER INTERFACES!



TRADITIONAL PAGES VS SPA



SO... A COMPONENT?

```
class App extends Component {
  render() {
    return (
      <div className="App">
        <h1>This is a component!</h1>
      </div>
    )
  }
}
```

JSX

```jsx
class App extends Component {
  render() {
    return (
      <div className="App">
        <h1>This is a component!</h1>
      </div>
    )
  }
}
```

```jsx
class App extends Component {
    render(){return React.createElement('div', {className: 'App'}, React.createElement('h1', null, 'This is a component!'))}
}
```

CLASS

```
class MyComponent extends Component {
  render() {
    return <div>Hello!</div>
  }
}
```

FUNCTIONAL

```
const MyComponent = () => {
  return <div>Hello!</div>
}
```

01

CLASS COMPONENTS

Props & State

CLASS COMPONENTS - PROPS

```tsx
import React, { Component } from 'react'

export class Planet extends Component<PlanetProps> {
  render() {
    const { name, children } = this.props

    return (
      <div>
        <h2>
          Planet {name} with satellites: {children}
        </h2>
      </div>
    )
  }
}

type PlanetProps = {
  name: string
}
```

```tsx
<Planet name={'Saturn'}>
  <h1>Titan</h1>
</Planet>
```

CLASS COMPONENTS – STATE

```tsx
export class PlanetarySystem extends Component<PlanetarySystemProps, PlanetarySystemState> {
  state = {
    planets: [{ name: 'Saturn' }, { name: 'Earth' }],
  }

  addPlanetHandler = () => {
    //DON'T DO THIS: this.state.planets.push({name: "New Planet"})
    const currentPlanets = this.state.planets
    this.setState({
      planets: [...currentPlanets, { name: 'New Planet' }],
    })
  }

  render() {
    return (
      <div>
        <h1>Universe</h1>
        <button onClick={this.addPlanetHandler}>Add planet</button>
        {this.state.planets.map(({ name }) => (
          <Planet name={name} />
        ))}
      </div>
    )
  }
}

type PlanetarySystemProps = {
  name: string
}

type PlanetarySystemState = {
  planets: { name: string }[]
}

```

02

FUNCTIONAL COMPONENTS

Props & State

FUNCTIONAL COMPONENTS - PROPS

```tsx
import React from 'react'

export const Astronaut = (props: { firstName: string; lastName: string }) => (
  <h2>
    Astronaut {props.firstName} {props.lastName}
  </h2>
)

```

```tsx
<Astronaut firstName={'Valentina'} lastName={'Tereshkova'} />
```

FUNCTIONAL COMPONENTS – STATE

```jsx
export const Mission = () => {
  const [astronautsState, setAstronautStates] = useState({
    astronauts: [
      { firstName: 'Valentina', lastName: 'Tereshkova' },
      { firstName: 'Yuri', lastName: 'Gagarin' },
    ],
  })

  const [spaceCraftState, setSpaceCraftState] = useState('')

  const addAstronaut = () => {
    setAstronautStates({
      astronauts: [...astronautsState.astronauts, { firstName: 'Juan', lastName: 'Martínez' }],
    })
  }

  const addSpaceCraft = () => {
    setSpaceCraftState('Vostok 1')
  }

  return (
    <div>
      <h1>Mission 1</h1>
      <button onClick={addAstronaut}>Add astronaut</button>
      <button onClick={addSpaceCraft}>Add spacecraft</button>
      <Astronaut
        firstName={astronautsState.astronauts[0].firstName}
        lastName={astronautsState.astronauts[0].lastName}
      />
      <Astronaut
        firstName={astronautsState.astronauts[1].firstName}
        lastName={astronautsState.astronauts[1].lastName}
      />
      {spaceCraftState !== '' && (
        <div>
          <p>Will travel on a</p>
          <Spacecraft name={spaceCraftState} />
        </div>
      )}
    </div>
  )
}
```

03

LIFECYCLE & HOOKS

CLASS-BASED VS FUNCTIONAL

LIFECYCLE VS HOOKS - EFFECTS 1

CLASS

```
componentDidMount() {
  // This code will run when mounting
}

componentDidUpdate() {
  // This code will run when updating
}
```

FUNCTIONAL

```
useEffect(()=>{
  // This code will run in every single render
})
```

RUN CODE IN EVERY SINGLE RENDER

LIFECYCLE VS HOOKS - EFFECTS 2

CLASS

```
componentDidMount(){
  // This code will run in the startup
}
```

FUNCTIONAL

```
useEffect(()=>{
  // This code will run in the startup
}, [])
```

RUN CODE ONLY IN FIRST RENDER

LIFECYCLE VS HOOKS - EFFECTS 3

CLASS

```
componentDidMount() {
  sideEffect()
}

componentWillUnmount(){
  cleanup()
}

```

FUNCTIONAL

```
useEffect(()=>{
  sideEffect()
  return function cleanup () {/*cleanup code*/}
}, [])

```

RUN A SIDE EFFECT ON FIRST RENDER AND CLEANUP

LIFECYCLE VS HOOKS - EFFECTS 4

CLASS

```
componentDidMount() {
  sideEffect()
}

componentDidUpdate(prevProps, prevState) {
  if (prevState.myThing !== this.state.myThing){
    sideEffect(this.state.myThing)
  }
}

componentWillUnmount(){
  cleanup()
}
```

FUNCTIONAL

```
useEffect(()=>{
  sideEffect(props.myThing)
  return function cleanup () {/*cleanup code*/}
}, [props.myThing])
```

RUN SIDE EFFECT IN EVERY RENDER (EFFICIENTLY)

04

ADVANCED USE OF THE STATE

PROBLEM WITH STATE?

PLANNING OUR SPACE TRIP

NUMBER OF SPACECRAFTS

```
handleChange(e) {
  this.setState({ numSpacecrafts: parseInt(e.target.value) })
}
<input onChange={this.handleChange} />
<p> { this.state.numSpacecrafts } </p>
```

AMOUNT OF FUEL

```
handleChange(e) {
  this.setState({ fuelAmount: parseInt(e.target.value) })
}
<input onChange={this.handleChange} />
<p> { this.state.fuelAmount } </p>
```

PROBLEM WITH STATE?

PLANNING OUR SPACE TRIP

NUMBER OF SPACECRAFTS

```
handleChange(e) {
  this.setState({ numSpacecrafts: parseInt(e.target.value) })
}
<input onChange={this.handleChange} />
<p> { this.state.numSpacecrafts } </p>
```

AMOUNT OF FUEL

```
handleChange(e) {
  this.setState({ fuelAmount: parseInt(e.target.value) })
}
<input onChange={this.handleChange} />
<p> { this.state.fuelAmount } </p>
```

SYNCED?

PROBLEM WITH STATE?

PLANNING OUR SPACE TRIP

SPACE TRIP PLANNER
(LOGIC TO KEEP THEM SYNCED)

```
1  <SpacecraftComponent numSpacecrafts={this.state.numSpacecrafts} onSpacecraftsChange={this.onSpacecraftChange}/>
2  <FuelComponent fuelAmount={this.state.fuelAmount} onFuelChange={this.onFuelChange}/>
```

NUMBER OF SPACECRAFTS

```
1  <input onChange={this.props.onSpacecraftsChange} />
2  <p>{this.props.numSpacecrats}</p>
```

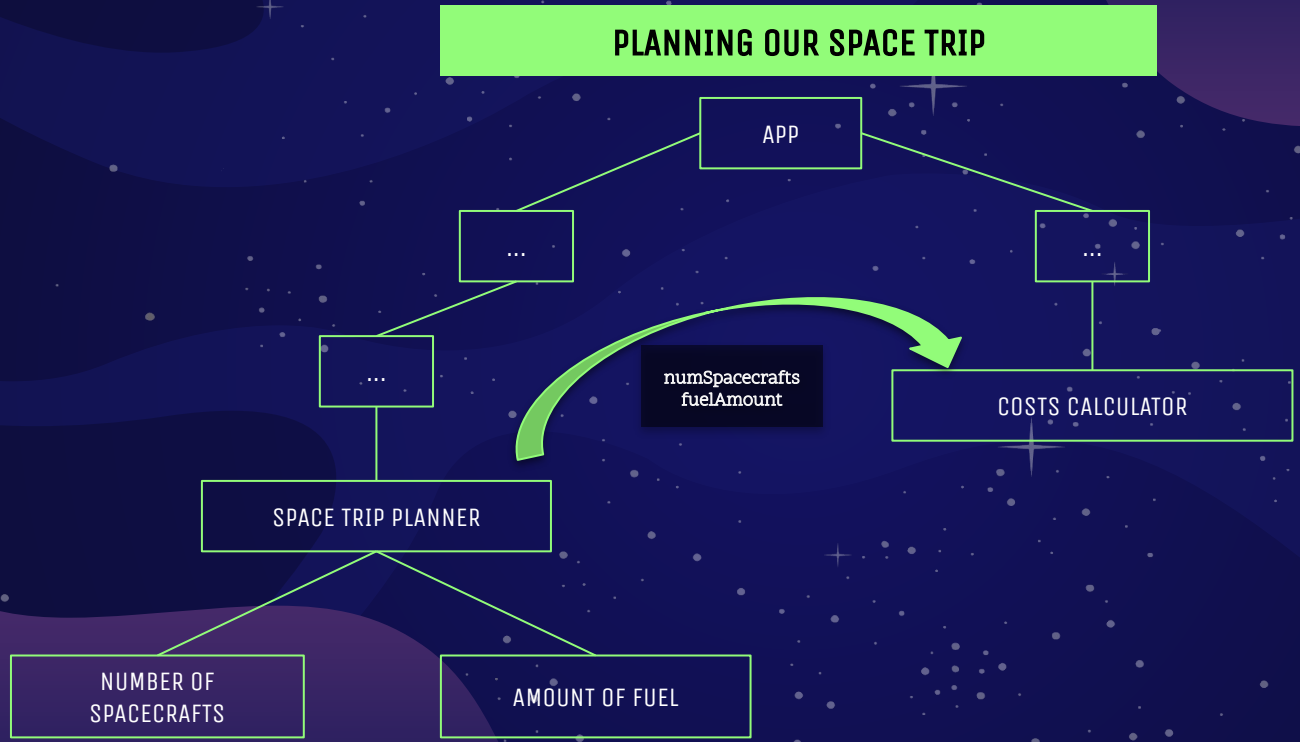
AMOUNT OF FUEL

```
1  <input onChange={this.props.onFuelChange} />
2  <p>{this.props.fuelAmount}</p>
```

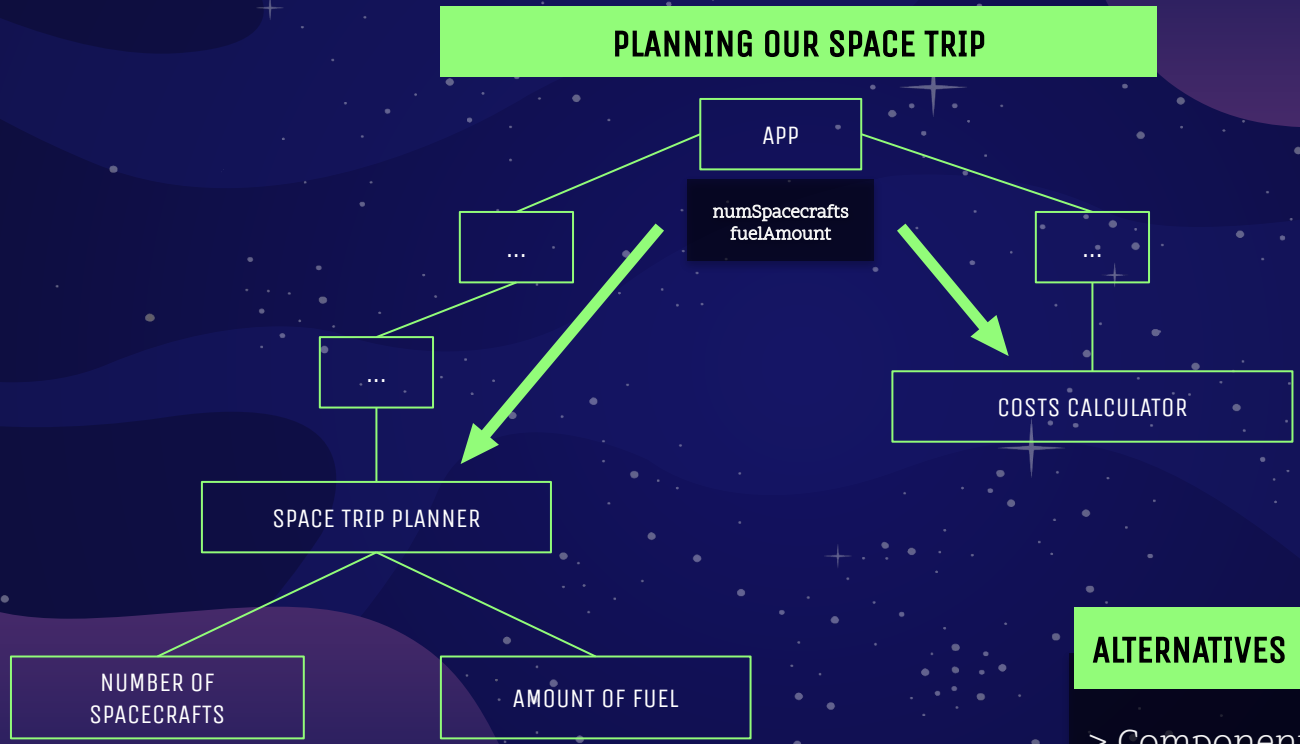
SYNCED!

- LIFTING STATE UP
- CONTROLLED COMPONENTS

PROBLEM WITH STATE?



PROBLEM WITH STATE?



ALTERNATIVES

> Component compositions
> Contexts
> Global store

GLOBAL STORE?

REDUX!

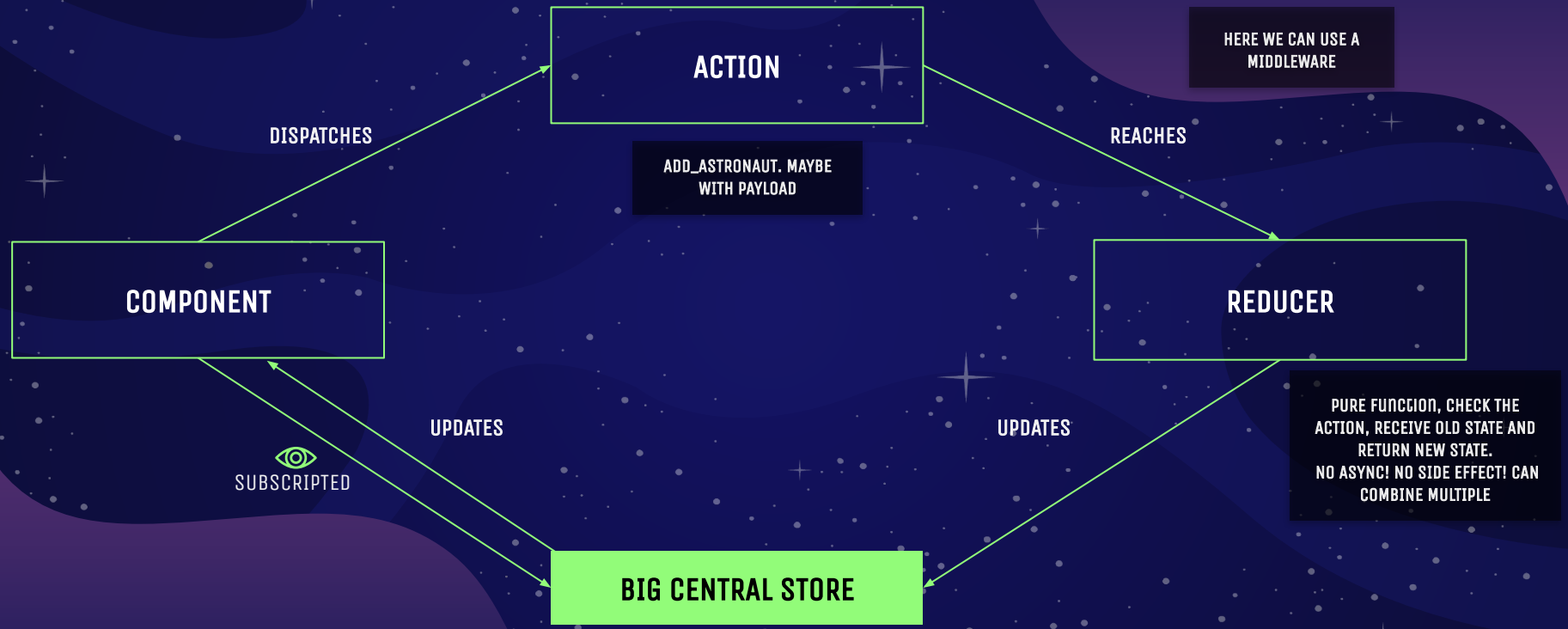
GLOBAL STORE?

REDUX!

ALTERNATIVES

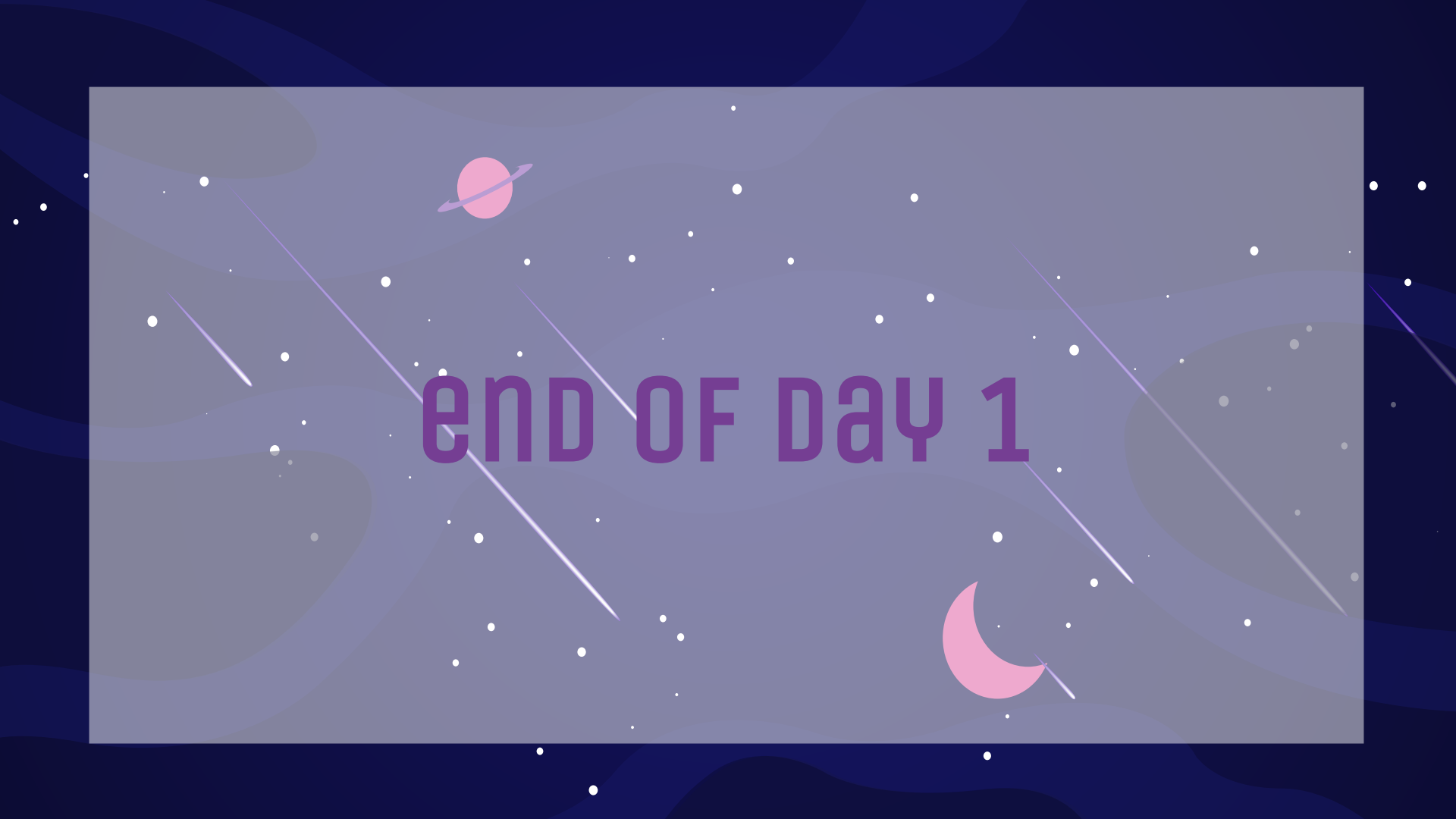
React Query is not a global store but may cover most of the use cases. We'll see that in a future talk :)

GLOBAL STORE? REDUX!



Redux

Let’s dive into it! : D


end of day 1

DAY 2 - WIP

References

- https://reactjs.org/docs/
- https://medium.com/javascript-in-plain-english/how-to-avoid-prop-drilling-in-react-using-component-composition-c42adfcdde1b
- https://www.freecodecamp.org/news/scaling-your-redux-app-with-ducks-6115955638be/
- https://medium.com/swlh/the-good-the-bad-of-react-redux-and-why-ducks-might-be-the-solution-1567d5bdc698
- https://github.com/erikras/ducks-modular-redux
- Lemon Coders channel in Youtube

Credits

- Presentation template by Slidesgo
- Icons by Flaticon
- Images & infographics by Freepik
- Author introduction slide photo created by Freepik
- Text & Image slide photo created by Freepik.com
- Big image slide photo created by Freepik.com