# React hands on workshop

By Vijay Shivakumar

# Welcome to the React workshop

Vijay Shivakumar

www.technicaltrainings.com

# What do we need before we begin... ?

**Technical Skill :** HTML5, CSS3, JavaScript 1.8.5

**Hardware and software:**

IDE : visual studio code
Browsers : chrome latest
Platform : nodejs latest
Database : mongodb
Version Control : git
Network : internet access to download from git and npmjs.org

# What do we need before we begin... ?

**Technical Skill :** HTML5, CSS3, JavaScript 1.8.5

**Hardware and software:**

IDE : visual studio code
Browsers : chrome latest
Platform : nodejs latest
Database : wamp / mongodb / mlab / firebase
Version Control : git
Network : internet access to download from git and npmjs.org

# Objectives

Understand and explore ES6 / ES7
Write Programs using Pure React
Understand JSX usage
Develop programs using React platform
Workflow with Context API
Workflow with Redux
Usage of middleware Saga
Unit testing with Jest

# What are we learning in this course ?

ES6+
Functional programming
Arrow functions
Immutable objects
Template strings
Destructuring
Array Methods
Scope Management
History API
CSS / SASS

What is React ?
Tooling and setup for React-CLI
React Components
Understanding JSX
Data binding
Class and Style binding
State
Props and PropTypes
Conditional Rendering
Working with Forms
Events

Context API
Lifecycle Methods
Working with HTTP
Provider API
Redux
React Routing
Lists and Keys
Fragments
Firebase / MongoDB

# Your Instructor / About Me

Vijay Shivakumar

Designer | Developer | Trainer

Adobe Certified Expert

Training & Consultation of
Contemporary Web Technologies and Adobe products from past 14 years

# Audience / About you

Developer
Designer
Manager
Architect
Technology Enthusiast

# Prerequisites / Before we begin

- HTML 5
- CSS3
- ES6
- NodeJS
  - TypeScript
  - WebPack
  - Express
  - Babel
- TDD
- MongoDB

# Must Know / Before we begin

HTML 5
CSS3

*I assume you know*


ES6

- block scope
- de structuring
- arrow function
- default parameters
- spread operator
- array methods
- template strings
- classes
- modules
- interfaces

# Must Know / Before we begin

HTML 5
CSS3

*I assume you know*


***We shall learn these***

ES6

- block scope
- de structuring
- arrow function
- default parameters
- spread operator
- array methods
- template strings
- classes
- modules
- interfaces

# About React / Why do we need react ?

Client Side Programming LIBRARY

Created and maintained by Facebook developers

Used to build dynamic user interfaces (Frontend)

Everything is a component

Often referred as V in the MVC

# What makes React great / Principles of React

DOM Manipulation only with React

Components architecture / Composability

One way data flow (Unidirectional Data Flow)

A solid UI library

# Why React / Advantages of React

## Component architecture

Easy to scale existing applications one component at a time

## Partial refresh of UI Virtual DOM

Only renders the area that is modified avoiding page refresh

## Fast

Client get a faster response from application and they are happy

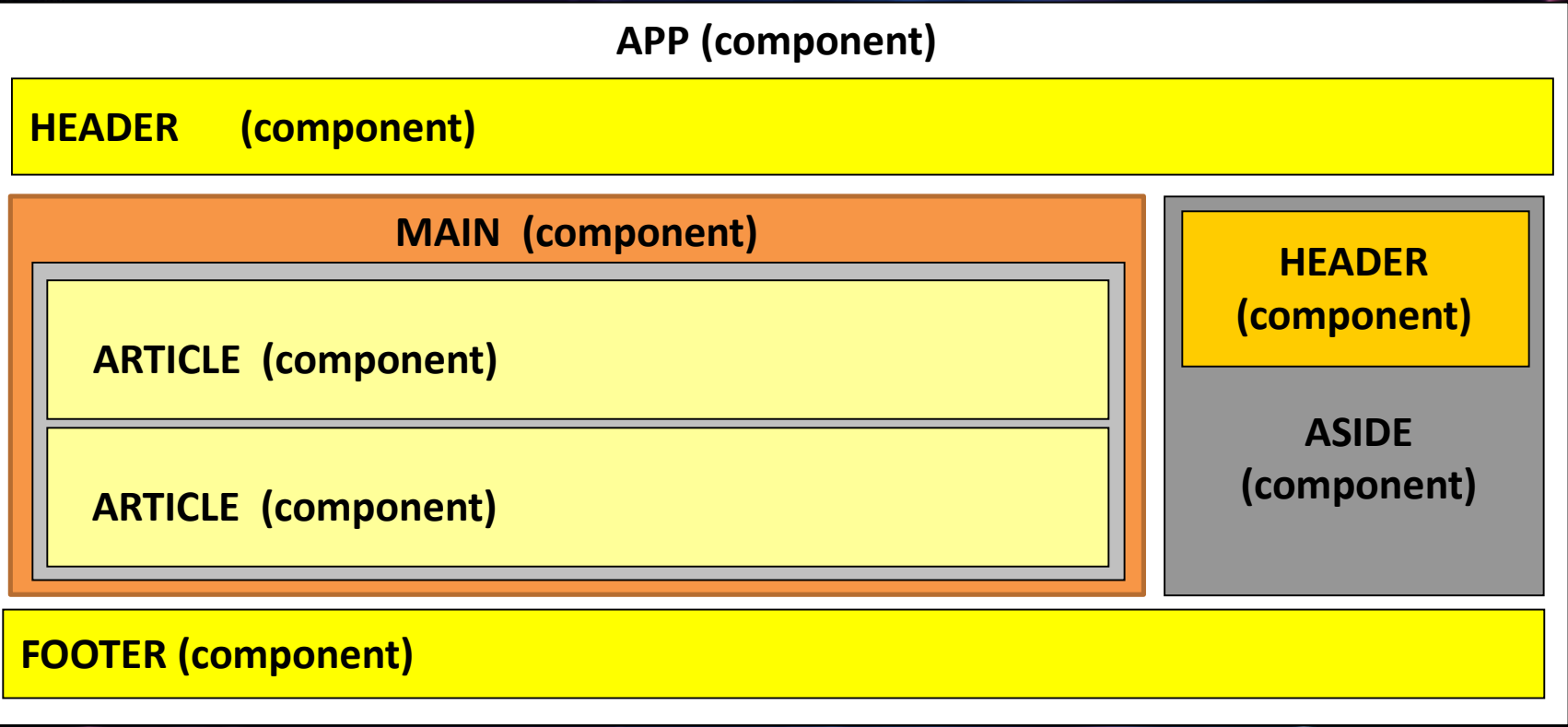
## Leverage on ES6 and later

Existing knowledge of JavaScript can be used to scale with react

# Benefits / Key take away from this course

- Components
  - Just like functions
  - Reusable and composable
  - Can manage a private state
- Reactive updates
  - Updates with user interaction
  - Take updates to the browser
- Virtual view in memory
  - write HTML in JavaScript
  - Tree reconciliation

# React Architecture



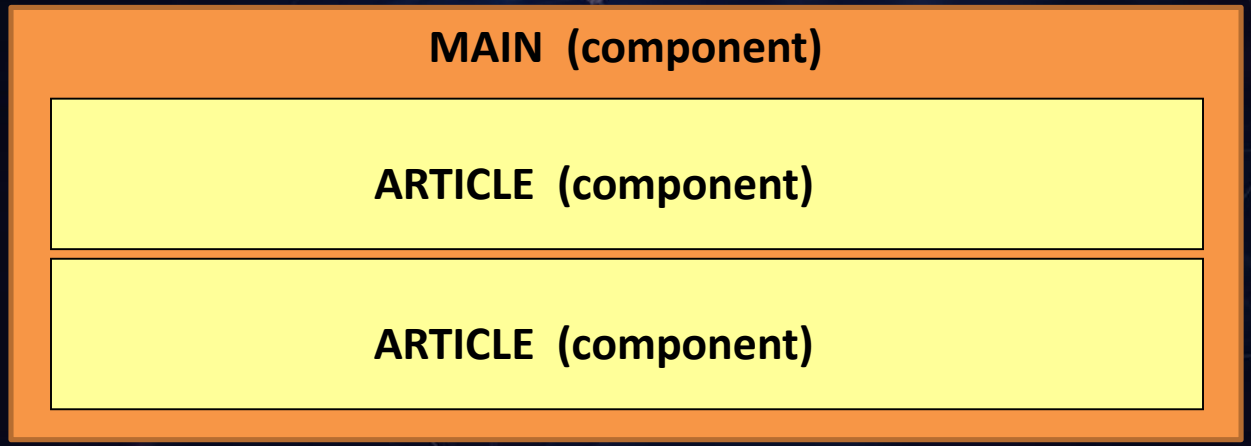
# JSX / Rules

1. JSX must return a single root element or a component
2. orphan tags needs to be closed eg <br/> <hr/> <img/> don’t use <br>
3. if you need to return multiple element and don't want to wrap them in a div use React.Fragment or <>;
4. intropolation using { 2 + 3 }
5. use className instead of class attribute
6. htmlFor instead of for attribute on input elements
7. use defaultValue instead of value on input element to create uncontrolled inputs
8. use camel cased so refer type for events for eg, onClick, onMouseOver
9. while using inline style use a config object
10. style properties should be camel cased
11. elements name begin with lowercase like h1, div, img, input, button etc
12. jsx classes, function names must begin with uppercase characters like Panel, UserForm, Datagrid etc

# Components

# Components / Building blocks of an application

## Composition



# Component / Types

Function components are stateless hence less dynamic

Class components are stateful and  are dynamic

# Redux

Vijay Shivakumar

# What is Redux ?

- Manages Data Store that can be accessed across you app
- Redux makes state management more predictable by having a single source or truth
- We can set strict rules for how the state can be updated

# What is Redux ?

A store — an immutable object that holds the applications state data

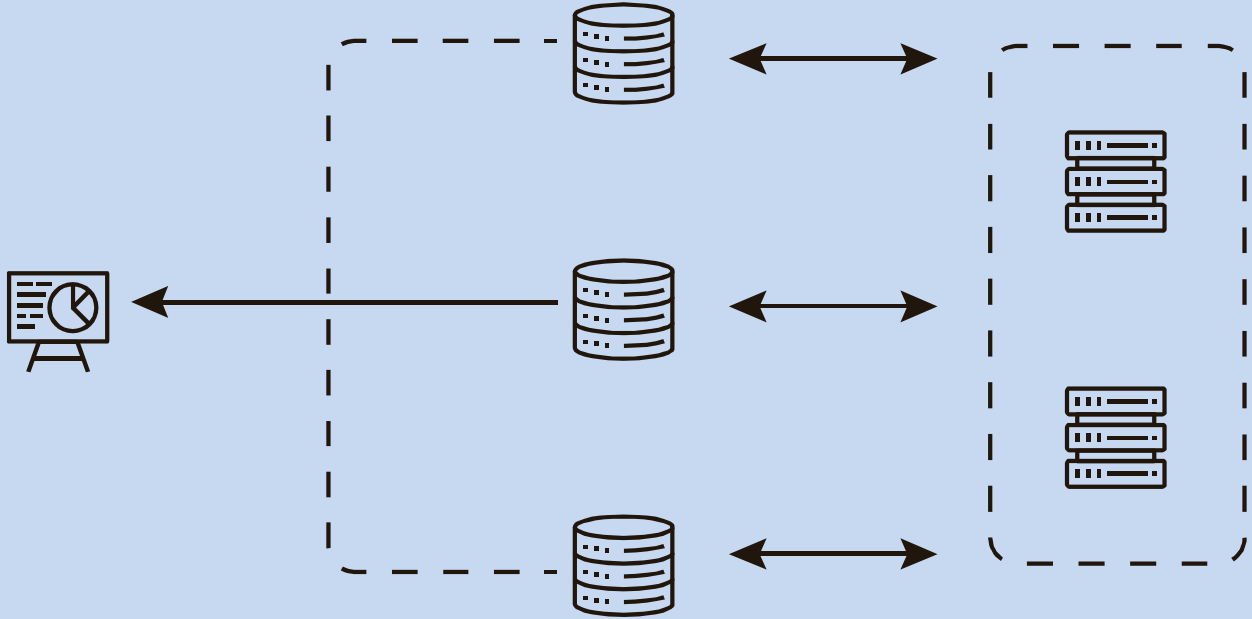
A reducer — a function that returns state data, triggered by an action type

An action — an object that tells the reducer how to change the state. It must contain a type property, and it can contain an optional payload property

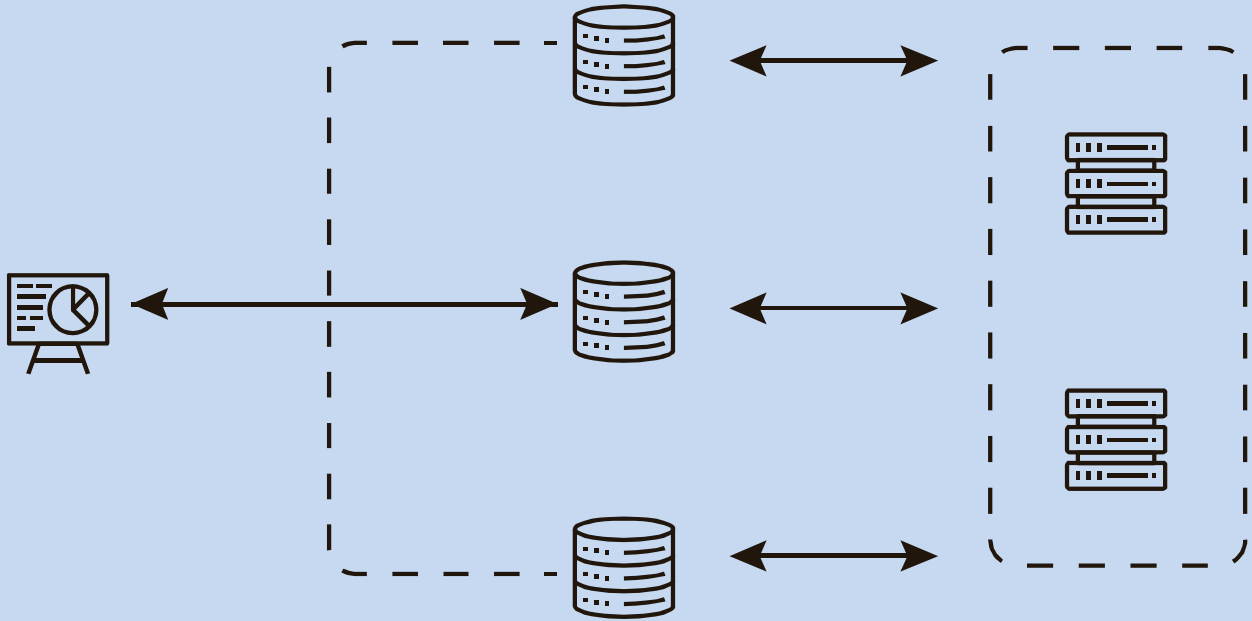
# Three Principles of Redux

- Application's state is stored in a single object tree which is managed by redux store.
- Only way to change the state is via a reducer function which knows what type of action happened.
  (should not update the state by any other means)
- How the state tree is transformed by action that are directed by reducer function

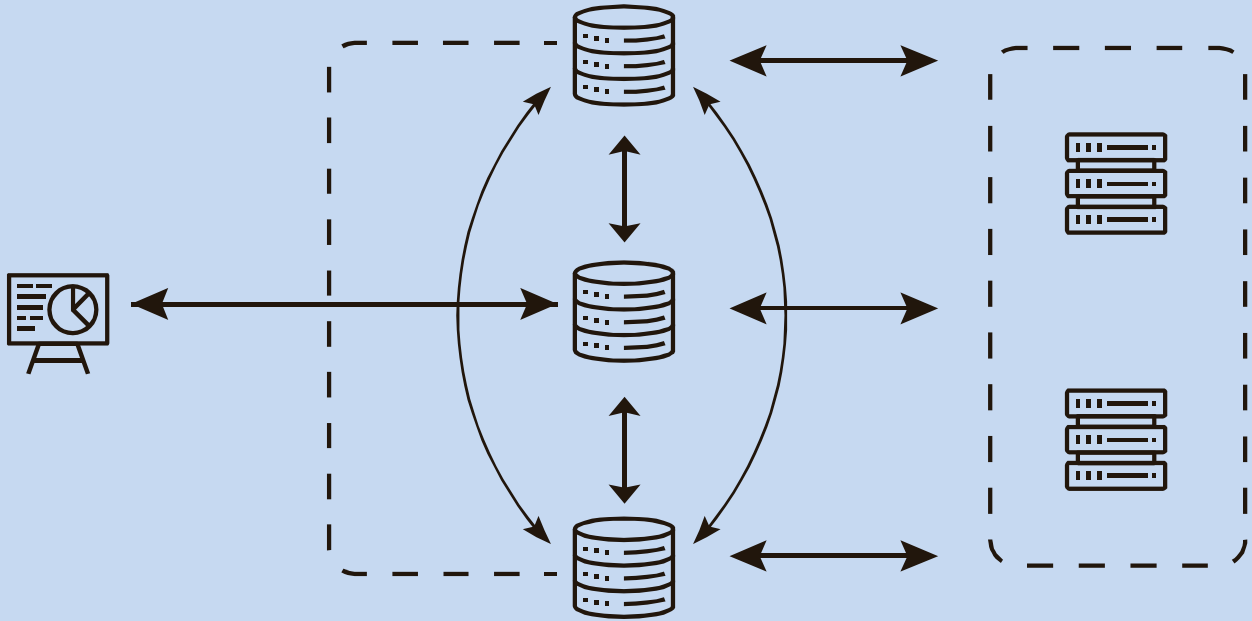
# Why Redux ?



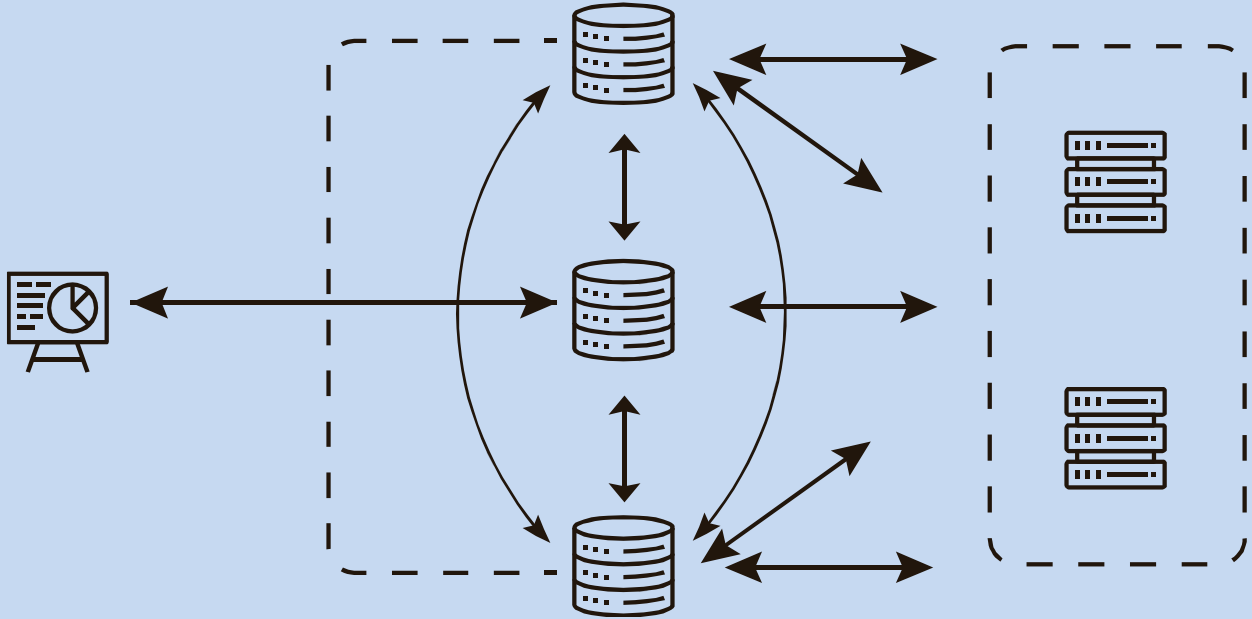
# Why Redux ?



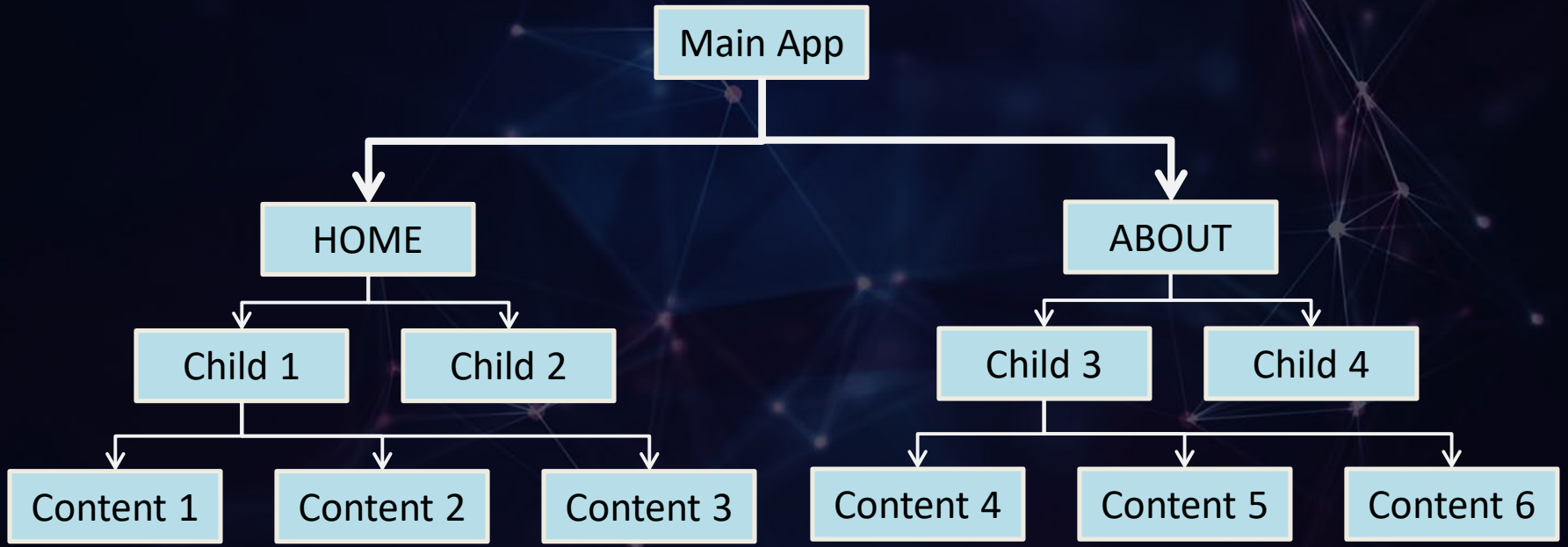
# Why Redux ?



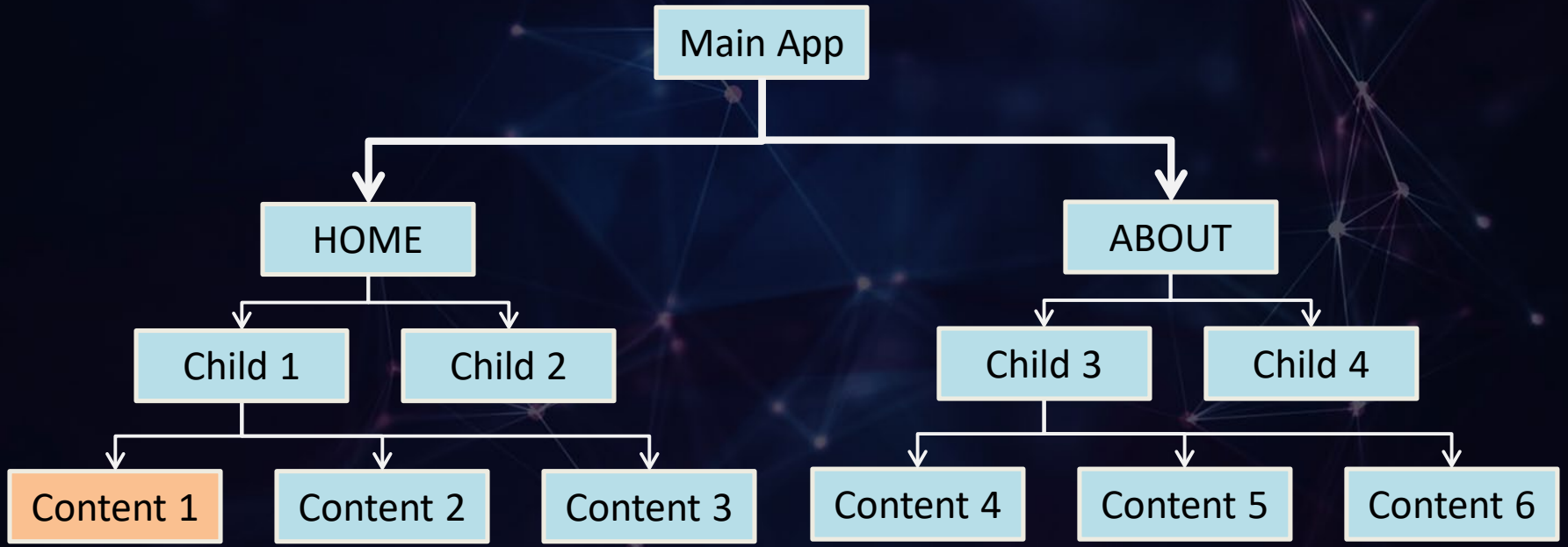
# Why Redux ?



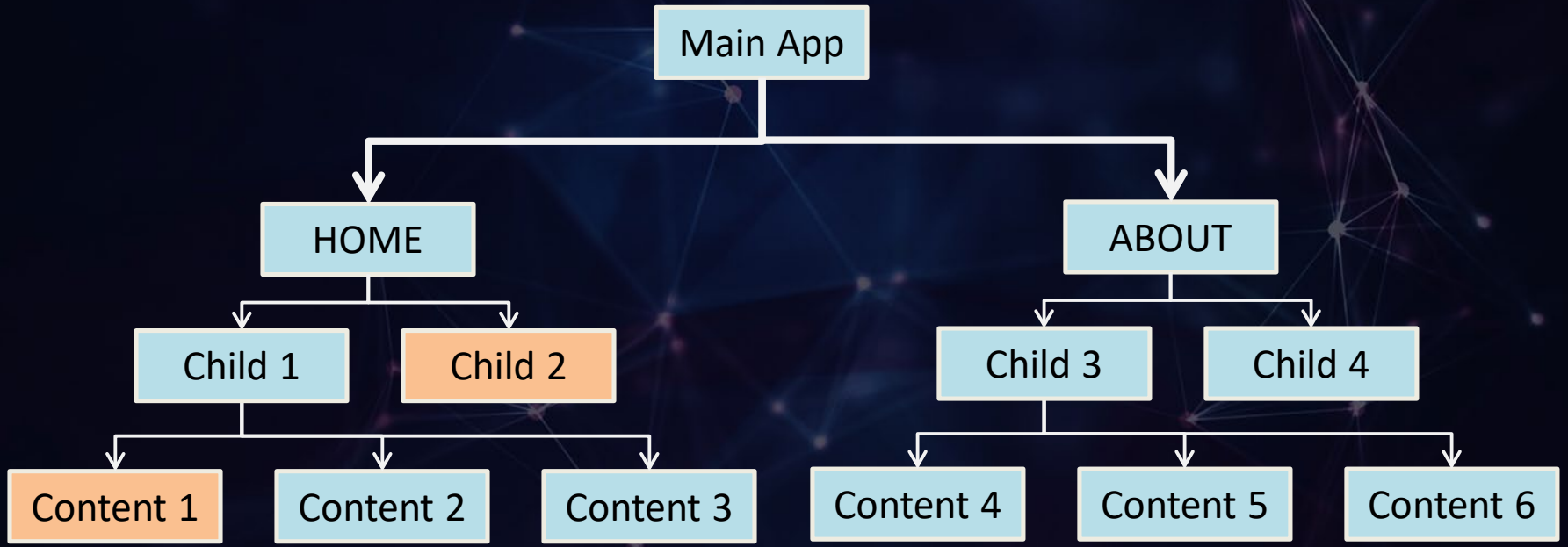
# Data flow issues



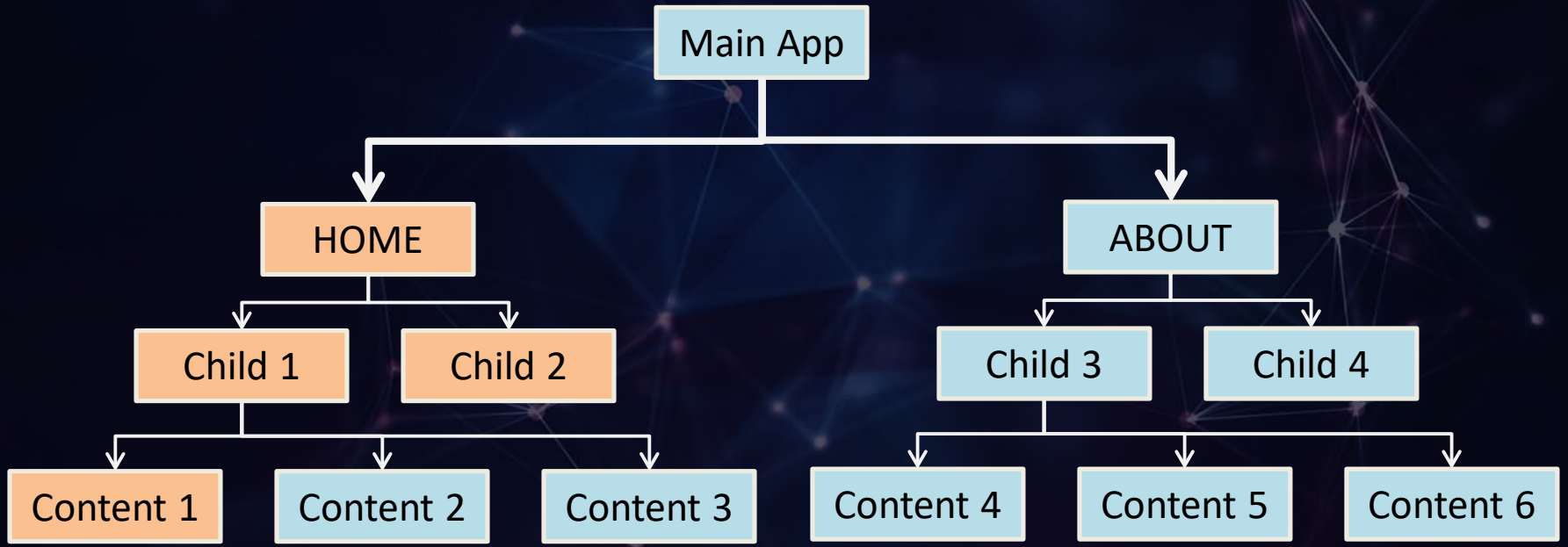
# Data flow issues



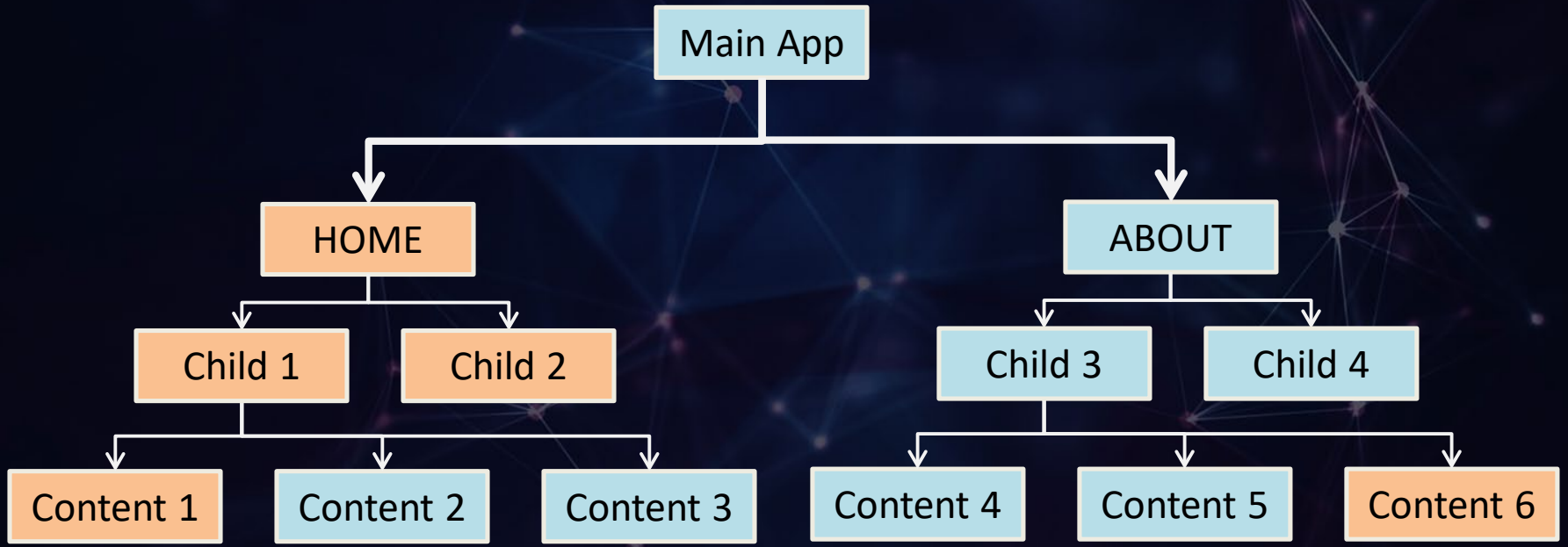
# Data flow issues



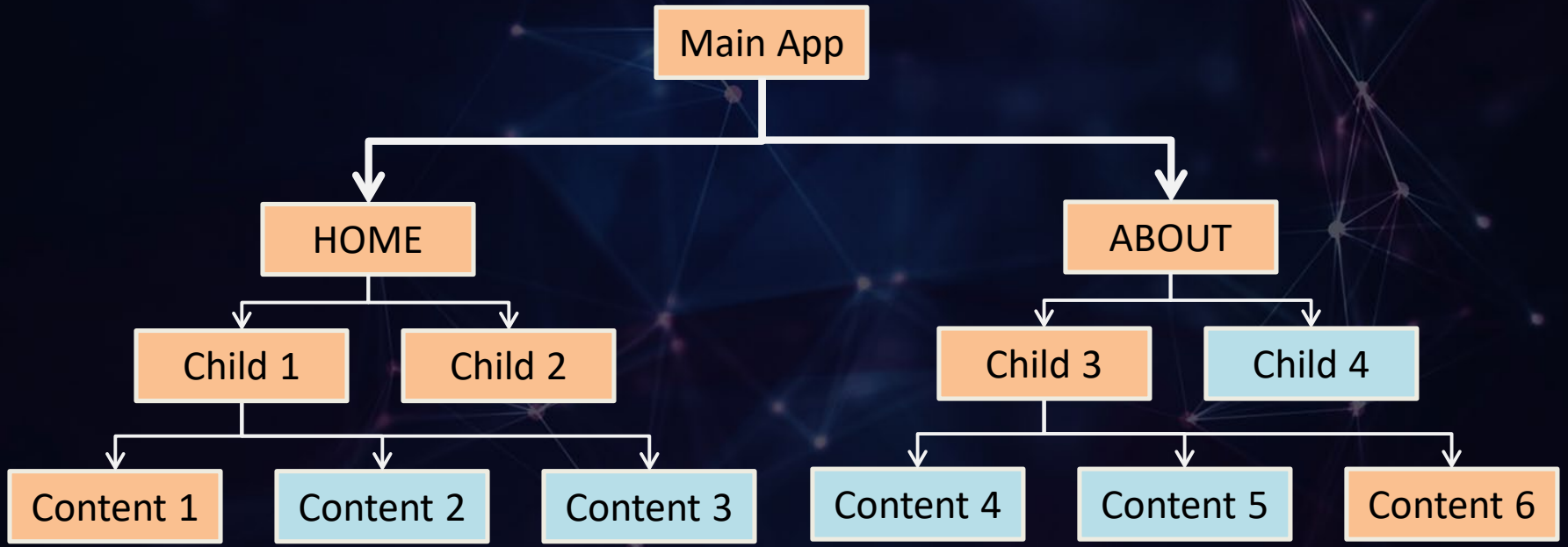
# Data flow issues



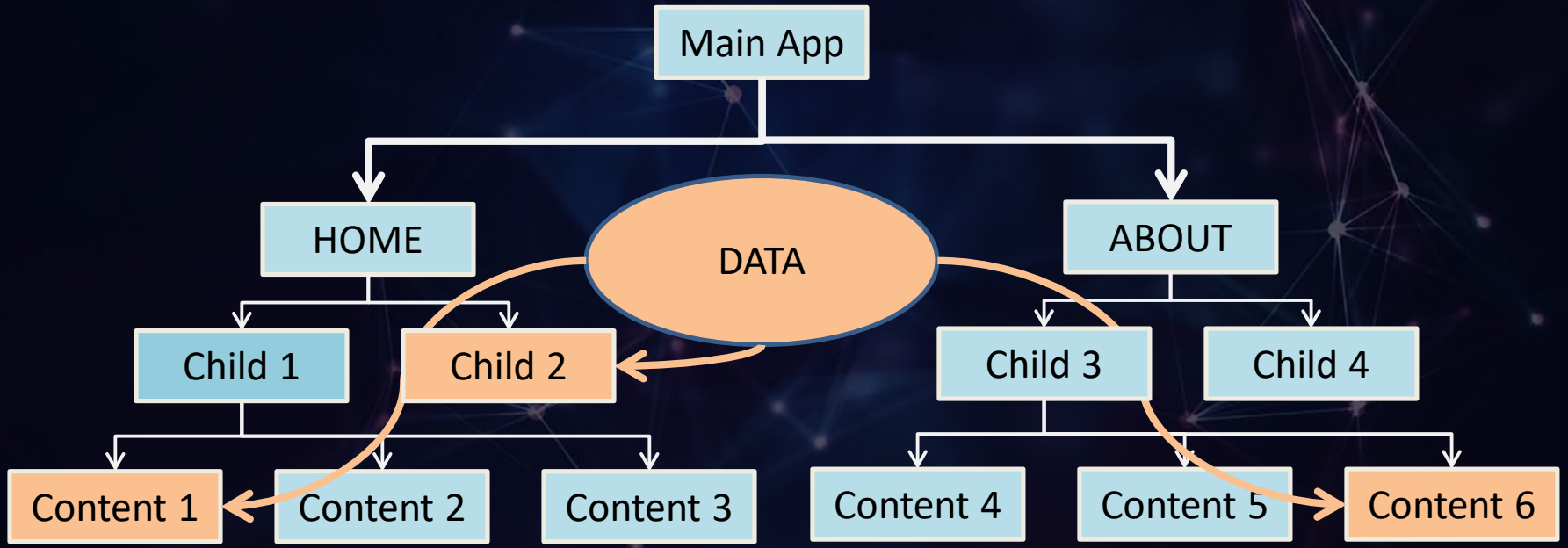
# Data flow issues



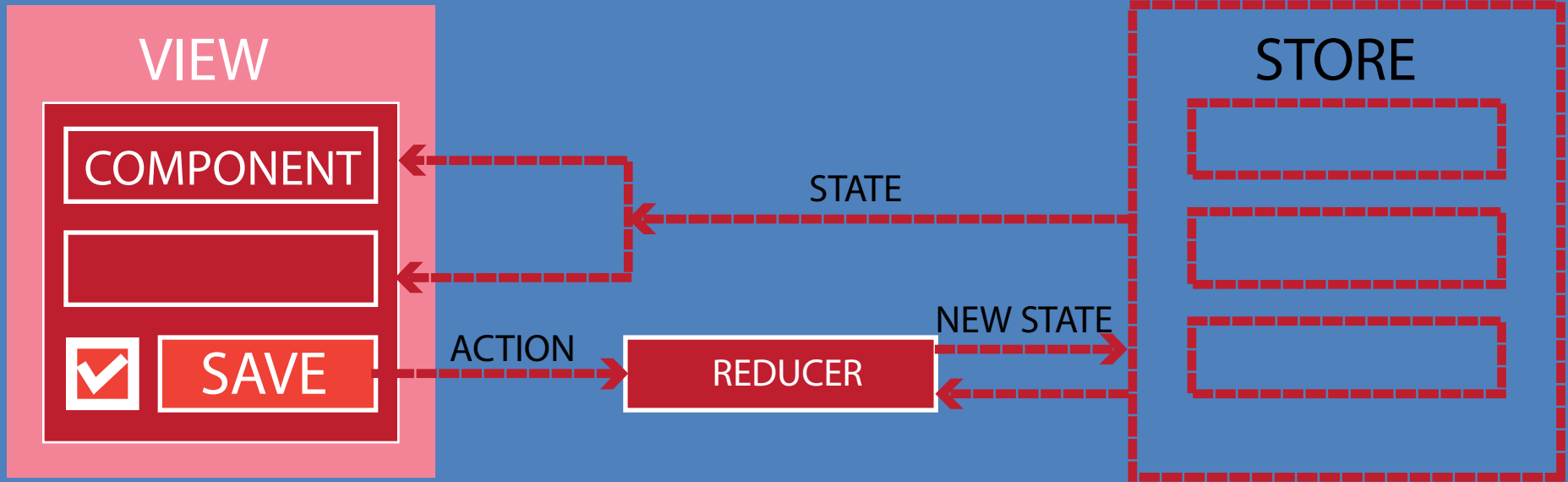
# Data flow issues



# Data flow issues



# Redux Architecture



Vijay Shivakumar

# Sagas

Vijay Shivakumar

# Introduction - Redux Saga

A Redux Middleware : Saga will be added in redux to cause an effect

Consumes Action : Saga consumes an action and in-turn may cause actions or side effects ( async operations )

Runs continuously : Maintains continuous running process called sagas

# What is Saga ?

A long running background process

Responsible for application's side effects

Ability to reverse the changes if failed

Leverages on ES6 generators and yield

It's a process manager Sagas manages one or more saga like starting or stopping them

Summery : Sagas listen to actions and dispatch other actions using effects which can modify external resources like databases file-system etc.

# Why Saga ?

## Helps in making side effects easy

API calls, database transactions

## Real world use cases

forking process (stop a process so another can run), yielding thread

## Better than Thunk

Thunk encourage putting lot of code in action creators and at times it becomes a mess

(disadvantage of saga is its learning curve)

# Thunk and Saga Comparision

| | |
|---|---|
| Thunk was developed by Redux developers | Developed by 3rd Party developers |
| Works in JavaScript | Works only on browsers that support ES5 with Yield |
| Issues between thunks when managing side effects between them | Uses plain actions to coordinate sagas |

# Testing

Vijay Shivakumar

vijay.shivu@gmail.com